



## Cortex AgentiX Documentation

Confidential - Copyright © Palo Alto Networks



## 1. Learn about Cortex AgentiX

- 1.1. Get Started with Cortex AgentiX
- 1.2. Understand Cortex AgentiX licenses
- 1.3. Agentic AI in Cortex AgentiX
  - 1.3.1. Manage Cortex AgentiX agents
  - 1.3.2. Cortex AgentiX use cases
  - 1.3.3. Agentic Assistant security
- 1.4. Key components and concepts
- 1.5. Data retention policy
- 1.6. Supported ciphers
- 1.7. Supported web browsers

## 2. Onboard Cortex AgentiX

- 2.1. Plan and prepare your deployment
- 2.2. Onboarding checklist
- 2.3. Activate Cortex AgentiX
  - 2.3.1. Cortex AgentiX supported regions
  - 2.3.2. Enable access to Palo Alto Networks resources
- 2.4. Install and configure content
- 2.5. Post-deployment steps
  - 2.5.1. Post-deployment checklist
  - 2.5.2. Set up users and roles
  - 2.5.3. Configure server settings
  - 2.5.4. Configure security settings
  - 2.5.5. Set up an engine
  - 2.5.6. Log forwarding
    - 2.5.6.1. Forward logs and data from Cortex AgentiX to external services
    - 2.5.6.1.1. Configure external applications for forwarding
    - 2.5.6.1.1.1. Integrate a syslog receiver
    - 2.5.6.1.1.2. Integrate Slack for outbound notifications
    - 2.5.6.1.2. Configure notification forwarding
    - 2.5.6.1.3. Monitor administrative activity
  - 2.5.6.2. Data and log notification formats
    - 2.5.6.2.1. Issue notification format
    - 2.5.6.2.2. Management Audit log notification format

## 3. Cortex AgentiX Configuration

- 3.1. Configure Cortex AgentiX
- 3.2. Content configuration
  - 3.2.1. Configure content using Data Sources
    - 3.2.1.1. Manage instances
    - 3.2.1.2. Add a new data source or instance
    - 3.2.1.3. Configure integration permissions
  - 3.2.2. Marketplace
    - 3.2.2.1. Cortex Marketplace
    - 3.2.2.2. Content packs
    - 3.2.2.3. Content Pack Support Types
    - 3.2.2.4. Cortex AgentiX content
    - 3.2.2.5. Manage content packs
    - 3.2.2.6. Marketplace FAQs
    - 3.2.2.7. Content changes when upgrading Cortex AgentiX versions
    - 3.2.2.8. Content pack contributions
  - 3.2.3. Integrations
    - 3.2.3.1. Manage API keys
    - 3.2.3.2. Integration use cases
    - 3.2.3.3. Add an integration instance
    - 3.2.3.4. Fetch issues from an integration instance
      - 3.2.3.4.1. Map fields to issue types
      - 3.2.3.4.2. Classify events using a classifier for issue types
    - 3.2.3.5. Forward Requests to Long-Running Integrations
    - 3.2.3.6. Manage credentials
  - 3.2.4. Troubleshoot Integrations
  - 3.2.5. Measuring data freshness
  - 3.2.6. Overview of data ingestion metrics
  - 3.2.7. Verify collector connectivity
  - 3.2.8. Creating correlation rules to monitor data ingestion health
- 3.3. Configure the Cortex Agentic Assistant
  - 3.3.1. Agentic Assistant components and concepts
  - 3.3.2. Agents Hub
    - 3.3.2.1. Manage actions
    - 3.3.2.2. Register actions
    - 3.3.2.3. Manage agents



- 3.3.2.4. Build agents
- 3.3.2.5. Expand agent capabilities with MCP integrations

### 3.3.3. Agentic Assistant role-based access control

## 3.4. Cortex MCP server

### 3.4.1. Cortex MCP server overview

- 3.4.1.1. Install the Cortex MCP server
- 3.4.1.2. Configure the MCP client
- 3.4.1.3. Use the Cortex MCP server
- 3.4.1.4. Create custom Cortex MCP server tools

## 3.5. Users and Roles Management

### 3.5.1. Users and roles in Cortex AgentIX

### 3.5.2. Roles management

- 3.5.2.1. Role-based permissions in Cortex AgentIX
- 3.5.2.2. Manage roles in the Cortex AgentIX tenant

### 3.5.3. User group management

### 3.5.4. User management

- 3.5.4.1. Manage users in the Cortex AgentIX tenant
- 3.5.4.2. Manage user scope

### 3.5.5. Manage access to objects

- 3.5.5.1. Manage access to custom dashboards
- 3.5.5.2. Manage access to saved queries

### 3.5.6. Set up authentication

- 3.5.6.1. Authenticate users through the Customer Support Portal
- 3.5.6.2. Authenticate users using SSO
- 3.5.6.3. Set up Okta as the Identity Provider Using SAML 2.0
- 3.5.6.4. Set up Azure AD as the Identity Provider Using SAML 2.0

## 3.6. Engines

### 3.6.1. What is an engine?

### 3.6.2. Engine requirements

### 3.6.3. Install an engine

#### 3.6.3.1. Docker

- 3.6.3.1.1. Install Docker
  - 3.6.3.1.1.1. Install Docker distribution for Red Hat on an engine server
  - 3.6.3.1.1.2. Docker image security
  - 3.6.3.1.1.3. Docker FAQs
  - 3.6.3.1.1.4. Troubleshoot Docker issues
  - 3.6.3.1.1.5. Configure Docker pull rate limit
  - 3.6.3.1.1.6. Change the Docker installation folder
- 3.6.3.1.2. Docker hardening guide

#### 3.6.3.2. Podman

- 3.6.3.2.1. Change the container storage directory
- 3.6.3.2.2. Install Podman
- 3.6.3.2.3. Migrate From Docker to Podman
- 3.6.3.2.4. Troubleshoot Podman

### 3.6.4. Manage engines

### 3.6.5. Upgrade an engine

### 3.6.6. Remove an engine

### 3.6.7. Configure engines

- 3.6.7.1. Configure the engine to use a web proxy
- 3.6.7.2. Configure the engine to call the server without using a proxy
  - 3.6.7.2.1. Use NGINX as a reverse proxy
- 3.6.7.3. Configure an engine to use custom certificates

### 3.6.8. Use an engine in an integration

### 3.6.9. Run a script using an engine

### 3.6.10. Troubleshoot engines

### 3.6.11. Troubleshoot integrations running on engines

## 3.7. Cases and issues configuration

### 3.7.1. Case and issue lifecycle

### 3.7.2. Customize your cases

- 3.7.2.1. External integrations
- 3.7.2.2. Create a case domain
- 3.7.2.3. Create a starring configuration
- 3.7.2.4. Create custom case statuses and resolution reasons

### 3.7.3. Customize issue fields and layouts

#### 3.7.3.1. Issue fields

- 3.7.3.1.1. Issue field types
- 3.7.3.1.2. Create custom issue fields
  - 3.7.3.1.2.1. Create a grid field for an issue
  - 3.7.3.1.2.2. Issue field triggered scripts
  - 3.7.3.1.2.3. Issue timer fields
    - 3.7.3.1.2.3.1. Configure issue timer fields
    - 3.7.3.1.2.3.2. Configure a playbook to run timers
    - 3.7.3.1.2.3.3. Automate changes to issue fields using timer scripts
    - 3.7.3.1.2.3.4. Use issue timer field commands manually in the CLI

#### 3.7.3.2. Issue layouts

- 3.7.3.2.1. Create custom issue layouts
- 3.7.3.2.2. Add a custom widget to an issue layout
- 3.7.3.2.3. Create rules for issue layouts

### 3.7.4. Customize case fields and layouts

#### 3.7.4.1. Case fields

- 3.7.4.1.1. Case field types
- 3.7.4.1.2. Create custom case fields
  - 3.7.4.1.2.1. Create a grid field for a case
  - 3.7.4.1.2.2. Update case fields
- 3.7.4.1.3. Create case timers and SLAs
- 3.7.4.1.4. Update case timer and SLA fields

#### 3.7.4.2. Case layouts

- 3.7.4.2.1. Create custom case layouts



#### 3.7.4.2.2. Create rules for case layouts

##### 3.7.5. Issue syncing

###### 3.7.5.1. Create a sync profile

##### 3.7.6. What's a correlation rule?

###### 3.7.6.1. Correlation rule details

###### 3.7.6.2. Create a correlation rule

###### 3.7.6.3. Manage correlation rules

##### 3.7.7. Manage external dynamic lists

#### 3.8. Automations

##### 3.8.1. Automation in Cortex AgentIX

##### 3.8.2. Quick Actions

##### 3.8.3. Automation Exclusion Center

###### 3.8.3.1. Manage automation exclusion policies

##### 3.8.4. Playbooks

###### 3.8.4.1. Playbooks overview

###### 3.8.4.2. Playbook development checklist

###### 3.8.4.3. Plan your playbook

###### 3.8.4.4. Manage playbooks

###### 3.8.4.5. Build your playbook

###### 3.8.4.5.1. Task 1. Choose from existing playbooks or create your own

###### 3.8.4.5.2. Task 2. Configure playbook settings

###### 3.8.4.5.3. Task 3. Add objects from the Task Library

###### 3.8.4.5.3.1. Add Quick Actions, commands, and scripts

###### 3.8.4.5.3.2. Add sub-playbooks

###### 3.8.4.5.3.3. Add AI prompt tasks

###### 3.8.4.5.3.4. Add manual tasks and blank tasks

###### 3.8.4.5.3.4.1. Create a standard task

###### 3.8.4.5.3.4.2. Create a conditional task

###### 3.8.4.5.3.4.3. Create a communication task

###### 3.8.4.5.3.5. Create a section header

###### 3.8.4.5.3.6. Configure script error handling in a playbook

###### 3.8.4.5.4. Task 4. Add custom playbook features

###### 3.8.4.5.5. Task 5. Test and debug the playbook

###### 3.8.4.5.6. Task 6. Manage playbook content

###### 3.8.4.6. Customize your playbook

###### 3.8.4.6.1. Configure a sub-playbook loop

###### 3.8.4.6.2. Filter and transform data

###### 3.8.4.6.2.1. Filter considerations, categories, and built-in filters

###### 3.8.4.6.2.2. Transformer considerations, categories, and built-in transformers

###### 3.8.4.6.3. Extract indicators

###### 3.8.4.6.4. Extend context

###### 3.8.4.6.5. Update issue fields with playbook tasks

###### 3.8.4.6.6. Playbook polling

###### 3.8.4.7. Test your playbook

###### 3.8.4.7.1. Troubleshoot playbook performance

###### 3.8.4.8. Manage playbook content

###### 3.8.4.9. Best practices

##### 3.8.5. Create an automation rule

##### 3.8.6. AI Prompts

###### 3.8.6.1. AI prompts role-based access control

###### 3.8.6.2. Use existing prompts

###### 3.8.6.3. Create a prompt

##### 3.8.7. Scripts

###### 3.8.7.1. Use existing scripts

###### 3.8.7.2. Create a script

###### 3.8.7.3. Use the Automation Engineer agent to accelerate script development and deployment

###### 3.8.7.4. Change the Docker image in an integration or script

##### 3.8.8. Context data

###### 3.8.8.1. Issue context data

###### 3.8.8.2. Case context data

###### 3.8.8.3. Search context data

###### 3.8.8.4. Add context data to an issue

###### 3.8.8.5. Add context data to a case

###### 3.8.8.6. Delete context data from a case

###### 3.8.8.7. Use context data in a playbook

#### 3.9. Jobs

##### 3.9.1. Manage jobs

##### 3.9.2. Create a time triggered job

##### 3.9.3. Create a job triggered by a delta in a feed

#### 3.10. Lists

##### 3.10.1. Create a list

##### 3.10.2. List commands

##### 3.10.3. Use cases: JSON lists

##### 3.10.4. Transform a list into an array

#### 3.11. Data management

##### 3.11.1. Dataset management

###### 3.11.1.1. What are datasets?

###### 3.11.1.2. Lookup datasets

###### 3.11.1.2.1. Import a lookup dataset

###### 3.11.1.2.2. Download JSON file of lookup dataset

###### 3.11.1.2.3. Set time to live for lookup datasets

###### 3.11.1.3. Monitor datasets and dataset views activity

##### 3.11.2. What are Data Model Rules?

###### 3.11.2.1. Data Model Rules editor views

###### 3.11.2.2. Data Model Rules file structure and syntax

###### 3.11.2.3. MODEL

###### 3.11.2.4. RULE



- 3.11.2.5. Field structure
  - 3.11.2.6. How to map authentication story events?
  - 3.11.2.7. Create Data Model Rules
  - 3.11.2.8. Troubleshooting Data Model Rules
  - 3.11.2.9. Using data enrichment
  - 3.11.2.10. Data Model Rules notifications
  - 3.11.2.11. Monitor Data Model Rules activity
- 3.11.3. Manage Event Forwarding

### 3.12. Dashboards and Reports

- 3.12.1. About dashboards
  - 3.12.1.1. Command Center dashboards
    - 3.12.1.1.1. AgentIX Command Center
  - 3.12.1.2. Predefined dashboards
- 3.12.2. Reports
  - 3.12.2.1. Report templates
  - 3.12.2.2. Run or schedule reports
- 3.12.3. Build custom dashboards and reports
  - 3.12.3.1. Build a custom dashboard
  - 3.12.3.2. Manage your Widget Library
- 3.12.4. Fine-tune dashboards and reports
  - 3.12.4.1. Create a custom widget using a script
    - 3.12.4.1.1. Script-based widget examples
  - 3.12.4.2. Create a text widget
  - 3.12.4.3. Create custom XQL widgets
    - 3.12.4.3.1. Configure filters and inputs for custom XQL widgets
  - 3.12.4.4. Configure dashboard drilldowns
    - 3.12.4.4.1. Variables in drilldowns

## 4. Investigation and Response

### 4.1. Overview of cases

- 4.1.1. What are cases?
- 4.1.2. Resolving cases with AI
- 4.1.3. Case lifecycle
- 4.1.4. Case thresholds
- 4.1.5. Case scope and impact
- 4.1.6. Case and issue domains

### 4.2. Case concepts

- 4.2.1. Issues, findings, and events
  - 4.2.1.1. Issues
  - 4.2.1.2. Findings and events
- 4.2.2. Case grouping
- 4.2.3. Case scoring
- 4.2.4. Case starring
- 4.2.5. SLAs and tracking

### 4.3. Analyze and resolve cases

- 4.3.1. Review all cases
- 4.3.2. Start case analysis
  - 4.3.2.1. Agentic Assistant- Case Investigation agent
- 4.3.3. Establish case context
  - 4.3.3.1. AI-generated case summaries
  - 4.3.3.2. Assess case severity and score
  - 4.3.3.3. Update case attributes
- 4.3.4. Analyze case details
  - 4.3.4.1. Grouping graph
  - 4.3.4.2. Evidence
  - 4.3.4.3. Issue feed
  - 4.3.4.4. Associated assets and artifacts
  - 4.3.4.5. MITRE ATT&CK tactics and techniques
  - 4.3.4.6. Detailed View
- 4.3.5. Resolve the case
  - 4.3.5.1. Resolution Center
  - 4.3.5.2. Collaborative notes and comments
  - 4.3.5.3. Resolve a case
  - 4.3.5.4. Resolution reasons for cases and issues
- 4.3.6. Use Cortex Agentic Assistant chat in an investigation
- 4.3.7. Create a case

### 4.4. Investigate issues

- 4.4.1. Overview of the Issues page
- 4.4.2. Issue card
- 4.4.3. Link or unlink issues from a case
- 4.4.4. Run an automation on an issue
- 4.4.5. Use the War Room in an investigation
- 4.4.6. Use the Work Plan in an investigation
- 4.4.7. Manage synced tickets in issues
- 4.4.8. Issue investigation actions
  - 4.4.8.1. Copy issues
  - 4.4.8.2. Investigate contributing events
  - 4.4.8.3. Export issue details to a file
  - 4.4.8.4. Exclude an issue



- 4.4.8.5. Query case and issue data
- 4.4.8.6. Run indicator extraction in the CLI
- 4.4.8.7. Update issue fields
- 4.4.8.8. Use issue timer field commands manually in the CLI
- 4.4.8.9. Close an issue

#### 4.5. Investigate artifacts and assets

- 4.5.1. Investigate an IP address
- 4.5.2. Investigate a file and process hash

### 5. Threat Intel Management

#### 5.1. Get started with Threat Intel Management

- 5.1.1. What is Threat Intel Management?
- 5.1.2. Threat Intel Management use cases
- 5.1.3. Roles and responsibilities in Threat Intel Management
- 5.1.4. Indicator concepts
- 5.1.5. Indicator lifecycle

#### 5.2. Indicator configuration

- 5.2.1. Customize indicator fields and types
  - 5.2.1.1. Create an indicator type
    - 5.2.1.1.1. Indicator type profile
    - 5.2.1.1.2. File indicators
    - 5.2.1.1.3. Formatting scripts
    - 5.2.1.1.4. Enhancement scripts
    - 5.2.1.1.5. Reputation scripts
    - 5.2.1.1.6. Reputation commands
    - 5.2.1.1.7. Map custom indicator fields
  - 5.2.1.2. Create an indicator field
    - 5.2.1.2.1. Indicator field structure
    - 5.2.1.2.2. Indicator field trigger scripts
- 5.2.2. Indicator classification and mapping
- 5.2.3. Indicator extraction
  - 5.2.3.1. Set the indicator extraction mode for a playbook task
  - 5.2.3.2. Disable indicator extraction for scripts or integrations
- 5.2.4. Configure indicator expiration
- 5.2.5. Configure Threat Intelligence feed integrations
- 5.2.6. Exclude indicators from enrichment
- 5.2.7. Generate issues from indicator rules
- 5.2.8. Export indicators

#### 5.3. Indicator management

#### 5.4. Indicator investigation

- 5.4.1. Indicator verdict
- 5.4.2. Extract and enrich an indicator
- 5.4.3. Expire an indicator
- 5.4.4. Manage indicator relationships
- 5.4.5. Delete and exclude indicators

### 6. Cortex AgentIX XQL

#### 6.1. Get started with XQL

- 6.1.1. XQL language features
- 6.1.2. XQL Language Structure
  - 6.1.2.1. Adding comments in queries
- 6.1.3. Supported operators
- 6.1.4. Datasets and presets
- 6.1.5. About examples
- 6.1.6. JSON functions
- 6.1.7. How to filter for empty values in the results table
- 6.1.8. Understanding string manipulation in XQL

#### 6.2. Build XQL queries

- 6.2.1. About the Query Builder
- 6.2.2. How to build XQL queries
  - 6.2.2.1. Get started with XQL queries
  - 6.2.2.2. Useful XQL user interface features
  - 6.2.2.3. XQL Query best practices
  - 6.2.2.4. Expected results when querying fields
  - 6.2.2.5. Create XQL query
  - 6.2.2.6. Review XQL query results
  - 6.2.2.7. Translate to XQL
  - 6.2.2.8. Graph query results
- 6.2.3. Query Builder templates
  - 6.2.3.1. Get started with Query Builder templates
  - 6.2.3.2. Considerations for using Query Builder templates
  - 6.2.3.3. Create a query from a template
  - 6.2.3.4. Run a free text query
  - 6.2.3.5. Query Builder template examples
- 6.2.4. Overview of the Query Center
  - 6.2.4.1. Edit and run queries in Query Center
    - 6.2.4.1.1. Query Center reference information
- 6.2.5. Manage scheduled queries



- 6.2.5.1. Scheduled Queries reference information
- 6.2.6. Manage your personal query library
- 6.2.7. Legacy Query Builder
  - 6.2.7.1. Create authentication query
  - 6.2.7.2. Create event log query
  - 6.2.7.3. Create file query
  - 6.2.7.4. Create image load query
  - 6.2.7.5. Create network connections query
  - 6.2.7.6. Create network query
  - 6.2.7.7. Create process query
  - 6.2.7.8. Create registry query
  - 6.2.7.9. Query across all entities

### 6.3. Stages

- 6.3.1. alter
- 6.3.2. arrayexpand
- 6.3.3. bin
- 6.3.4. call
- 6.3.5. comp
- 6.3.6. config
  - 6.3.6.1. case\_sensitive
  - 6.3.6.2. timeframe
  - 6.3.6.3. max\_runtime\_minutes
- 6.3.7. dedup
- 6.3.8. fields
- 6.3.9. filter
- 6.3.10. getrole
- 6.3.11. iploc
- 6.3.12. join
- 6.3.13. limit
- 6.3.14. replacenull
- 6.3.15. search
- 6.3.16. sort
- 6.3.17. Tag
- 6.3.18. target
- 6.3.19. top
- 6.3.20. transaction
- 6.3.21. union
- 6.3.22. view
- 6.3.23. windowcomp

### 6.4. Functions

- 6.4.1. add
- 6.4.2. approx\_count
- 6.4.3. approx\_quantiles
- 6.4.4. approx\_top
- 6.4.5. array\_all
- 6.4.6. array\_any
- 6.4.7. arrayconcat
- 6.4.8. arraycreate
- 6.4.9. arraydistinct
- 6.4.10. arrayfilter
- 6.4.11. arrayindex
- 6.4.12. arrayindexof
- 6.4.13. array\_length
- 6.4.14. arraymap
- 6.4.15. arraymerge
- 6.4.16. arrayrange
- 6.4.17. arraystring
- 6.4.18. avg
- 6.4.19. coalesce
- 6.4.20. concat
- 6.4.21. convert\_from\_base\_64
- 6.4.22. count
- 6.4.23. count\_distinct
- 6.4.24. current\_time
- 6.4.25. date\_floor
- 6.4.26. divide
- 6.4.27. earliest
- 6.4.28. extract\_time
- 6.4.29. extract\_url\_host
- 6.4.30. extract\_url\_pub\_suffix
- 6.4.31. extract\_url\_registered\_domain
- 6.4.32. first
- 6.4.33. first\_value
- 6.4.34. floor
- 6.4.35. format\_string
- 6.4.36. format\_timestamp



6.4.37. if  
6.4.38. incidr  
6.4.39. incidr6  
6.4.40. incidrlst  
6.4.41. int\_to\_ip  
6.4.42. ip\_to\_int  
6.4.43. is\_ipv4  
6.4.44. is\_known\_private\_ipv4  
6.4.45. is\_ipv6  
6.4.46. is\_known\_private\_ipv6  
6.4.47. json\_extract  
6.4.48. json\_extract\_array  
6.4.49. json\_extract\_scalar  
6.4.50. json\_extract\_scalar\_array  
6.4.51. json\_path\_extract  
6.4.52. lag  
6.4.53. last  
6.4.54. last\_value  
6.4.55. latest  
6.4.56. len  
6.4.57. list  
6.4.58. lowercase  
6.4.59. ltrim, rtrim, trim  
6.4.60. max  
6.4.61. median  
6.4.62. min  
6.4.63. multiply  
6.4.64. object\_create  
6.4.65. object\_merge  
6.4.66. parse\_epoch  
6.4.67. parse\_timestamp  
6.4.68. pow  
6.4.69. rank  
6.4.70. regexcapture  
6.4.71. regextract  
6.4.72. replace  
6.4.73. replex  
6.4.74. round  
6.4.75. row\_number  
6.4.76. split  
6.4.77. stddev\_population  
6.4.78. stddev\_sample  
6.4.79. string\_count  
6.4.80. subtract  
6.4.81. sum  
6.4.82. time\_frame\_end  
6.4.83. timestamp\_diff  
6.4.84. timestamp\_seconds  
6.4.85. to\_boolean  
6.4.86. to\_epoch  
6.4.87. to\_float  
6.4.88. to\_integer  
6.4.89. to\_json\_string  
6.4.90. to\_number  
6.4.91. to\_string  
6.4.92. to\_timestamp  
6.4.93. uppercase  
6.4.94. values  
6.4.95. var  
6.4.96. wildcard\_match

## 7. Troubleshoot

### 7.1. About health issues

- 7.1.1. Monitor data ingestion health
- 7.1.2. Monitor correlation rules
- 7.1.3. Investigate and resolve health issues

### 7.2. Log forwarding

- 7.2.1. Forward logs from Cortex AgentX to external services
  - 7.2.1.1. Configure external applications for forwarding
    - 7.2.1.1.1. Integrate a syslog receiver
    - 7.2.1.1.2. Integrate Slack for outbound notifications
  - 7.2.1.2. Configure notification forwarding
  - 7.2.1.3. Monitor administrative activity
- 7.2.2. Data and log notification formats
  - 7.2.2.1. Issue notification format



7.2.2.2. Management Audit log notification format

### 7.3. In-product support case creation

## 8. Reference

- 8.1. Cortex AgentX API
- 8.2. XDM fields for mapping authentication events
- 8.3. Fair Usage policy for Cortex AgentX



# 1 | Learn about Cortex AgentiX

## Abstract

Learn how Cortex AgentiX works, review licenses, service limits, and other key details.

Before diving in, understand Cortex AgentiX functionality and how it integrates with your needs.

## 1.1 | Get Started with Cortex AgentiX

### Abstract

Learn about Cortex AgentiX and the key integrated capabilities.

#### What is Cortex AgentiX?

Cortex AgentiX enables SecOps teams to orchestrate system and custom agents to plan and execute complex workflows, governed by enterprise-grade security and permissions management. Built upon our industry-leading security automation platform, Cortex AgentiX is designed for the era of AI agents, enhancing automations with AI-driven intelligent, dynamic operations. This revolutionary platform empowers security teams to leverage AI and automations to resolve significantly more cases at scale with minimal effort, unlocking unparalleled efficiency.

Cortex AgentiX can tap into thousands of proven integrations and automations to power dynamic, autonomous operations.



The Cortex AgentiX Command Center assists SOC teams in understanding how their organization utilizes Cortex AgentiX and its impact on improving Key Performance Indicators (KPIs) and overall security outcomes by providing a complete view of the Cortex AgentiX ecosystem, its agents, and executed plans. Its users can access information such as total triggers, agent plans, user prompts, interactions, as well as open cases.

#### Why Cortex AgentiX?

Cortex AgentiX has the following benefits:



- Improved SOC outcomes

Leveraging Security Orchestration, Automation, and Response (SOAR) capabilities with Agentic AI significantly improves Security Operations Center (SOC) outcomes by combining the strengths of both rule-based and AI-driven automation. While traditional rule-based automation efficiently handles routine security issues, ensuring consistent and rapid response for known threats, AI agents provide dynamic, on-demand assistance for ad-hoc or complex problems that don't fit predefined workflows, leading to faster resolution of novel threats.

- Intelligent investigation and response

Cortex AgentiX leverages intelligent automation and orchestration through agents to optimize security operations. Existing platform artifacts such as playbooks, scripts, and commands are transformed into actions. Users prompt agents to create and execute multi-step dynamic and responsive plans to augment day-to-day SOC operations.

- Enterprise-grade security and governance

Agents are bound by the same rules and robust permissions as a human user. In addition, you can mark actions that make real-world changes in production systems as sensitive, requiring a quick manual review and confirmation, ensuring peace of mind before critical system changes are made.

- System and custom agents for personalized assistance

Cortex AgentiX offers system agents that are mission-focused, as well as the ability to create custom agents. An analyst focused on threat hunting might work primarily with the system Threat Intel agent, while analysts focused on general investigations might build custom agents, including all the actions required to perform their daily tasks. Analysts no longer need to look up specific commands or switch between screens. They can prompt AgentiX in natural language prompts to find the information they require, to prompt the creation of plans, and to authorize sensitive actions, as needed.

- AI capabilities to enhance existing rule-based automation

Leveraging natural language, the user can ask any question, automate any task, and dramatically lower the bar for building automation using:

- **Automation Engineer agent to develop scripts:** Create fully functional Python automation scripts based on natural language prompts in the Agentic Assistant. Save time, reduce the need for coding expertise, and help maintain consistent coding standards.
- **AI prompts in playbooks:** You can add AI prompts to playbooks, enabling automated interaction with an LLM as a single step in a playbook. AI prompts contain inputs and outputs that guide the LLM to perform specific actions and provide structured results. For example, you can use an AI prompt to identify malware categories.

- Flexible and extendable SOAR capabilities

- **Marketplace:** The Cortex AgentiX Marketplace provides users with pre-built automation and orchestration content. Marketplace content packs contain AI actions, integrations, playbooks, dashboards, fields, and more, to support specific security orchestration use cases.
- **Customization:** With custom scripts, playbooks, fields, and layouts, you can build your own solutions for a wide variety of use cases, not limited to traditional SOC workflows.

- Improved investigation collaboration

Collaborative investigation features provide a powerful toolkit to help analysts assist each other, run real-time security commands, and learn from each issue with auto-documentation of all actions. An ML-driven assistant learns from actions taken in the platform and offers guidance on analyst assignments and commands to execute actions.

## 1.2 | Understand Cortex AgentiX licenses

### Abstract

The Cortex AgentiX license is downloaded from Cortex Gateway and determines which components users can use and how many users can access the tenant.

Cortex AgentiX requires a yearly license per user. Multi-year licenses are available.

### License usage

This table describes the types of Cortex AgentiX licenses which are used in the following circumstances:

Version	Usage	Description
Cortex AgentiX Enterprise	Built for customers who need a complete security automation solution, including threat intel management.	Cortex AgentiX SOAR with threat intel includes 4 users, six months of hot storage for incidents, and 800 compute units per year.



Version	Usage	Description
Cortex AgentiX Base	Built for security operations and incident response customers who need case management with collaboration and playbook-driven automation.	Cortex AgentiX SOAR includes 2 users, six months of hot storage for incidents, and 400 compute units per year.

#### Development/Production tenants

In Cortex AgentiX you can use a content management system with a remote repository to develop and test content. If you want a development tenant, you need a development tenant license. The development tenant license allows many users to build, test, and refine automations. It is not limited to your purchased user licenses, and it enables multiple stakeholders to work on these tasks. This supports faster innovation, more reliable workflows, and scalable solutions as your organization grows.

#### License quota

The following table describes the license quotas of each version in Cortex AgentiX.

	Cortex AgentiX Base	Cortex AgentiX Enterprise
Users	Two users, additional users can be added	Four users, additional users can be added
Compute units	400 compute units per year*	800 compute units per year*
Integrations	Unlimited	Unlimited
Incident Management	180-day history*	180-day history*
Incident Triggered Automations	Unlimited	Unlimited
Job Triggered Automations	Unlimited	Unlimited
Intel Feeds	5 active feeds, 100 indicators/fetch	Unlimited
Threat Intel Library	Intelligence detail view and relationship data are not included	Unlimited

#### NOTE:

\*\*You can purchase additional compute units.

\*You can extend incident retention by purchasing an add-on. For more information, see [Data retention policy](#).

Intel feed quotas are based on the selected Fetches Indicators field in the integration instance settings, not the enabled status. Disabling an integration instance does not affect the intel feed quota. For example, if the AWS Feed is enabled and is fetching indicators and you don't want to include this in your quota, open the integration settings and clear the Fetches Indicators checkbox.

#### Cortex AgentiX users

Cortex AgentiX has the following users:

##### Audit user

Audit users have read-only permission in Cortex AgentiX, meaning they cannot edit system components and data or run commands, scripts, and playbooks. Audit users can view incidents, dashboards, and reports.

##### Full user

Full users have read-write permission in Cortex AgentiX, meaning they can view and edit system components and data. They can investigate incidents, run scripts and playbooks, chat in the War Room, and more. Full users' access to Cortex AgentiX is determined by their assigned role.



## 1.3 | Agentic AI in Cortex AgentiX

### Abstract

Use the Cortex Agentic Assistant to investigate cases, perform threat hunting, and create scripts. Embed and run LLM prompts in playbooks. View AI case summaries.

Cortex AgentiX integrates advanced artificial intelligence to streamline security operations. Through the Cortex Agentic Assistant, the platform provides a unified interface for interacting with both system-provided and custom AI agents capable of creating and executing multi-step plans. These agents leverage specific capabilities to perform actions across your infrastructure, facilitating deep case investigations and proactive threat hunting while allowing for the creation of tailored automation.

### Key AI Capabilities

- **Agents Hub:** A centralized hub for managing agents and actions. System agents can be enabled and disabled, and you can create custom agents tailored to your organizational needs, including the ability to execute custom scripts.
- **Automation Engineer Agent:** Provides a natural language interface to draft, refine, and deploy automation scripts.
- **MCP Integration:** Supports the configuration of integrations that communicate with external MCP servers, enabling agents to access third-party tools and data sources via a standardized protocol.
- **Embedded AI Prompts:** Facilitates the inclusion of generative AI tasks within playbooks. These prompts function as standalone workflow steps to analyze data or generate content without requiring a dedicated agent.
- **AI-Generated Case Summaries:** Automatically generate technical overviews of security incidents. These summaries consolidate complex telemetry and impact data into high-level reports to accelerate initial triage and stakeholder reporting.

### 1.3.1 | Manage Cortex AgentiX agents

#### Abstract

Learn about personal and system agents in Cortex AgentiX.

In Cortex AgentiX, you can interact with agents in the Cortex Agentic Assistant chat to automate case and issue investigation and response. Agents create and execute plans, which are sequences of actions (such as playbooks, scripts, and commands) designed to fulfill users' requests.

#### Agent selection

Each agent is designed with specific goals and functions to help you address different aspects of security operations.

Within the chat prompt, click the agent icon to select available agents. As you hover over each agent, you can see a brief description of the agent and its primary focus. Select the agent that best suits your current task or investigation.

You can select from system agents, public agents other users have created, or agents you have personally built and configured.

Here are some examples of the specialized system agents you may encounter, each relevant for specific security workflows:

Agent Type	Description
IT	Automates identity lifecycle enforcement, real-time containment on endpoints and networks, vulnerability and patch governance, asset intelligence upkeep, and end-to-end incident workflow coordination—delivering policy-driven remediation across the enterprise.
Email Investigation	Automates the full lifecycle of email-borne threat response, spanning mailbox search, forensic collection, analysis, containment, and incident closure across all major mail platforms and security layers.
Threat Intel	Gathers fresh threat data, enriches indicators and vulnerabilities, links them to past or current incidents, and publishes clear briefings so the whole SOC acts on the latest attacker tactics.
Help Center	Access Palo Alto Networks' product documentation for additional product support.
Network Security	Audits next-gen firewalls for vulnerabilities, expired certificates, outdated software, risky or unused rules, capacity limits, and other misconfigurations. It searches logs for threats and then automates or guides clean-ups and upgrades to keep the network secure.



Agent Type	Description
EndPoint Investigation	Unifies host-level containment, forensic collection, and remediation across all major EDR/XDR platforms while feeding evidence and status into the SOC's ticketing and collaboration stack.
Automation Engineer	Streamlines automation by generating and updating Python scripts from natural-language prompts, applying security best practices, and letting you explore or explain any part of the code.

#### Agent Management

Both actions and agents are managed in the Agents Hub (go to Agentic Assistant in the main menu and click the side panel icon  to view the Agents Hub menu item). In the Agents Hub, you can do the following:

- Register scripts, commands, and AI prompts as custom actions. Items that are registered as custom actions can be assigned to agents and used in plans. For more information, see [Manage actions](#).
- View, enable, and disable system actions and edit existing custom actions.
- Enable and disable system agents. System agents have access to system actions that are assigned to the agent.
- Create, edit, and delete custom agents. Custom agents can include system actions as well as custom actions. Custom agents can be private for a user or available to all users.

#### NOTE:

When users create custom agents, the agents have the same or fewer permissions as the user. For example, if a user has sufficient permissions to isolate an endpoint, the user's agent can also isolate an endpoint.

All users have access to all system agents and all public agents, but plan execution is limited by the permissions of the individual user. A system or public agent may include actions that the user does not have permission to execute.

For more information and to configure actions and agents, see [Agents Hub](#).

### 1.3.2 | Cortex AgentiX use cases

#### Abstract

Recommended prompts to automate your SOC in Cortex AgentiX.

Discover how Cortex AgentiX can streamline your security operations by exploring some key use cases.

#### Chat prompt examples

Using chat prompt conversation starters in Cortex AgentiX simplifies and speeds up your interactions by providing pre-defined, common queries that guide you to relevant actions and information.

For example, a SOC analyst may see the following conversation starters under the chat prompt:

- What are the top issues I should prioritize today?
- Show me all issues with an overdue SLA
- Which automations are waiting for my input?
- Clean up all expired Indicators

Additional examples of possible relevant prompts are:

- Read this Unit42 blog and get all the CVEs. For every critical CVE found, check if my assets are vulnerable and isolate them.
- List recent security issues with high severity and an affected hostname that includes 'server'.
- Summarize the latest security issues from the past 24 hours
- How do I make a loop inside a playbook?
- What is the riskiest unresolved issue affecting our critical infrastructure?
- Show recent SSO-related issues
- Investigate this phishing issue and determine the source of the email and block any malicious indicators.



## AI prompt task examples

The following are use cases for incorporating an AI prompt task in a playbook to help standardize and scale processes and streamline case handling to increase analyst productivity.

- Threat summary

Use the Issue Summary and Remediation Recommendations system AI task that analyzes issue logs and indicators to generate a concise summary for the security analyst, saving manual review time.

- Malware analysis

Use an AI task that extracts insights from a sandbox malware report, filters noise, and generates a readable summary, identifying malware family traits, behavior, and extracting Indicators of Compromise (IoCs). This summary can be used in an automated workflow to quickly assess threats. It can help to quickly assess threats without manually reading extensive logs, and a summary can be automatically sent to Slack, email, or ITSM, such as ServiceNow.

- Vulnerability prioritization

Use an AI task to send a vulnerability scan report and have the associated agent summarize and prioritize the CVEs found in the scan report.

### 1.3.3 | Agentic Assistant security

#### Abstract

Learn about how the Agentic Assistant is built using responsible AI principles.

The Agentic Assistant is built on responsible AI principles to ensure its use is safe, fair, and trustworthy. We design our AI to be transparent about its actions, accountable for its decisions, and fair in its operations, avoiding biases.

The following describes how the Agentic Assistant protects sensitive data and gives you control and understanding over its automated actions.

#### Access control and permissions

##### User roles and RBAC options

Instance and Account admins have full control over the permissions and access that users have to the Cortex Agentic Assistant. Cortex AgentiX uses Role-Based Access Control (RBAC) to manage access to the chat, as well as access to view, create, edit, delete, disable, and enable Agents and Actions in the Agents Hub.

#### Action Execution Scope

Agents can only use actions that have been assigned to them, and execution is limited to the user's existing permissions in your Cortex AgentiX tenant. If a required integration is not active, its commands and any actions that wrap them will not work.

#### Data security and control

##### How sensitive data is protected

Data is hosted and encrypted by default on a dedicated Google Cloud Platform (GCP) project, and is isolated and protected by your specific IAM permissions. Google's multi-tenant architecture enforces strict data separation between customers.

#### User approval for sensitive actions

Actions marked as sensitive require explicit user approval before execution and are never run automatically. This gives you final control over critical or data-modifying steps.

#### Data user policy

Your prompts and outputs are processed only to generate the immediate response. They are not collected for model training or shared with third parties.

#### Data residency

All prompts and responses stay inside that region's compute boundary, aligning with modern data-residency practices.

#### Transparency

##### How the Cortex Agentic Assistant maintains transparency

You can see how the agent reaches its answer. Click the down arrow next to Plan, to view how the user input was interpreted, the planned steps, and the actions used. You can view JSON artifacts created during the plan execution, when data was retrieved or an object was created.

In addition, all actions an agent takes are saved in an audit dataset. You can see which agent ran which action, and which user invoked it.



## 1.4 | Key components and concepts

### Abstract

Learn about the key components and concepts, such as agents and actions in Cortex AgentiX.

Cortex AgentiX uses the following components and concepts:

Name	Description
Actions	<p>Actions wrap diverse capabilities (such as playbooks, scripts, and commands, and AI prompts) to make them accessible and executable by an agent. You can use out-of-the-box system actions or register new actions.</p>
Agent	<p>An agent is a virtual persona that creates and executes domain specific plans, at your request, to assist in your day-to-day SOC operations. An agent has roles and permissions that provide guardrails. Each agent is assigned a collection of actions that it can use as part of plans.</p> <p>The agent chooses the most relevant actions to fulfill a user's request. Agents process user requests, create plans, and orchestrate actions based on their goals and permissions (RBAC and SBAC).</p> <p>You can use the following types of agents:</p> <ul style="list-style-type: none"><li>• System agents that are provided by Cortex AgentiX for specific use cases.</li><li>• Custom agents that the user has created.</li></ul> <p>Some agents provide relevant chat conversation starters under the chat prompt. For examples of conversation starters, see Cortex AgentiX use cases.</p> <p><b>NOTE:</b></p> <p>Agents are bound by the same rules and robust permissions as a human user. In addition, you can mark actions that make real-world changes in production systems as sensitive, requiring a quick manual review and confirmation, ensuring peace of mind before critical system changes are made.</p>
Automation Rules	Automation rules allow the system to automatically respond to events by defining trigger conditions and desired actions, such as playbooks and Quick Actions, to perform once the condition is met.
Quick Actions	Quick Actions are preset single commands that enable you to automate basic tasks such as creating tickets in third-party systems, sending Slack messages, and changing issue severity.
Playbook	Playbooks are a series of tasks that run in a predefined flow. Playbooks can include sub-playbooks, manual tasks, automated tasks that run scripts or commands, AI tasks, and Quick Actions.
Issue	Issues identify the problems that you need to solve in your environment. Cortex AgentiX creates issues when problems occur in your environment that cross defined thresholds, or surpass your organization's accepted level of risk and threat tolerance.
Case	A case is a workbench for resolving security problems in your environment. Each case groups related issues, highlights the impacted assets, and provides essential data in one place. Cases help you stay focused on threats and risks that have the most impact on the organization's security, help you reduce noise in your environment, and guide you to resolution using automation actions that reduce time and effort.
Plan	A sequence of actions that run in parallel or sequentially to satisfy a user request. The agent dynamically chooses relevant actions to resolve the prompt.
Conversation	A sequence of user requests that maintain context across interactions.
Request	A user request from the agent with an end goal, triggering a plan.



## 1.5 | Data retention policy

### Abstract

Learn more about the default retention periods for all Cortex AgentiX licenses and the available retention add-ons.

Cortex AgentiX data storage is managed in the Cortex AgentiX Data Layer. You receive data storage based on the amount of storage associated with your licenses. Generally, this capacity is determined by factors such as your daily ingestion needs and the number of users in your deployment. All Cortex AgentiX licenses provide you with default retention periods. You can extend your license retention depending on your requirements for hot storage.

The following table summarizes the default retention periods for Cortex AgentiX:

Data Type	Default Retention Period	Notes
Ingested data	31 days	
Cases and Issues data	Six months	Cases data is retained according to the Last Updated date. Issue data is retained according to the Observation Time date. Data collected within these dates is kept and displayed for six months. To ensure the accuracy of issues, Cortex AgentiX provides a grace period of up to 31 days for issues displayed in the Issues view, Issues table, and Cases view.

For more information on your storage license details, see [Dataset Management](#).

### Retention add-ons

Retention add-ons are provided for ingested data and cases and issues data. Minimum requirements are dependent on the license type. You can purchase one or more of the following retention add-ons:

Feature	Description
Additional cases and issues retention	An additional 31-day hot storage of case and issue data apart from the default six months.
Period-based retention - hot storage	Fully searchable storage for investigation and threat hunting of ingested data, and cases and issues data. Requires purchasing a minimum of one month of the additional retention.
Additional hot storage	Additional hot storage for AgentiX base and enterprise incidents beyond the default.

With a regular Cortex AgentiX license, the data is automatically sent to hot storage for the default retention period according to the license, usually one month. If you've purchased additional retention add-ons to extend the hot storage, they are added in monthly increments to the hot storage duration according to the license.

For example, a Period-Based Retention - Hot Storage license enables you to extend all data in hot storage for the number of months designated. An Additional Hot Storage license provides flexible hot storage so you can extend only the data collected in specific datasets for the number of months designated. During the additional hot storage retention period, data continues to be sent from the Data Ingestion Pipeline. This data is no longer accessible after the hot storage retention period ends and the data is gradually purged.

## 1.6 | Supported ciphers

Cortex AgentiX supports the following Cipher Suites for TLS1.2 and TLS1.3:



- TLS ECDHE ECDSA with AES256 GCM SHA384
- TLS ECDHE RSA with AES256 GCM SHA384
- TLS ECDHE ECDSA with AES128 GCM SHA256
- TLS RSA with AES128 GCM SHA256

## 1.7 | Supported web browsers

Cortex AgentiX supports the following web browsers:

Browser	Version
Chrome	95.x and later
Firefox	93.x and later
Safari	13.x and later
Microsoft Edge	81.x and later
Prisma Browser	Latest version

### NOTE:

Cortex AgentiX is optimized for best performance using the HTTPv2 protocol. If you are using a non-supported browser version or a network middleware that uses a protocol other than HTTPv2, you may experience slowness in loading and response times between the browser and the Cortex AgentiX service.

Consult with your browser or middleware vendor to see how to update the protocol.

## 2 | Onboard Cortex AgentiX

### Abstract

Follow the steps to successfully onboard and configure Cortex AgentiX.

Get up and running quickly. Our intuitive Onboard section guides you through essential setup steps like activation and content management. Once you're set up, customize Cortex AgentiX to match your requirements.



# Cortex AgentiX Onboarding

This infographic outlines the essential journey for deploying Cortex AgentiX, guiding users through the critical stages from initial planning and preparation to core activation and post-deployment customization. Following these steps ensures a smooth and effective setup.

## Phase 1: Plan & Prepare



### Determine Your Deployment Strategy

Decide on your hosting region, initial data sources, and required system or custom agents.



### Plan User Access & Permissions

Review and plan Role-Based Access Control (RBAC) for your security operations team.

## Phase 2: Onboard & Activate



### Activate Your Tenant

Log in to Cortex Gateway to activate your tenant and define its name and region.



### Configure Core Content

Use the Data Sources wizard to ingest priority data and enable a threat intelligence feed.



### Set Up Initial Users

Assign administrator and analyst roles to a limited number of initial users.

## Phase 3: Post-Deployment



### Configure & Customize

Apply automation rules, customize server settings, and configure security policies.



### Extend & Integrate

Install content packs and deploy an engine for integrations with on-premise resources.



### Monitor & Operate

Review dashboards to visualize activity and set up log forwarding to external services.

## 2.1 | Plan and prepare your deployment

### Abstract

Before deploying your tenant, consider your use case and what you need to optimize your tenant.

Before you start your Cortex AgentiX deployment, consider the following:

Action	Details	See More
Determine the deployment region	Determine the region you want to host Cortex AgentiX and any associated services.	Cortex AgentiX supported regions
Review your license and add-ons	Review your Cortex AgentiX license and consider the addons for your use case, such as Threat Intel Management.	
Consider the data sources to use	Consider the data sources you want to ingest initially.  In Cortex AgentiX content is organized into content packs, which are either downloaded from the Data Sources catalog or from Marketplace. Start planning what content you require.  Review the steps you need to take in your day-to-day SOC operations, and the required third-party tools/applications.	Install and configure content
Consider the agents to use	Consider which agents you require? Review the system agents. Download related content packs, if needed, to make system actions available for system agents.  Consider whether to create new custom agents and what actions they would perform.	Manage Cortex AgentiX agents
Consider roles and permissions	Review and plan roles using Role-Based Access Control (RBAC) for your security operations team. Consider user groups and start with the default roles.	Set up users and roles

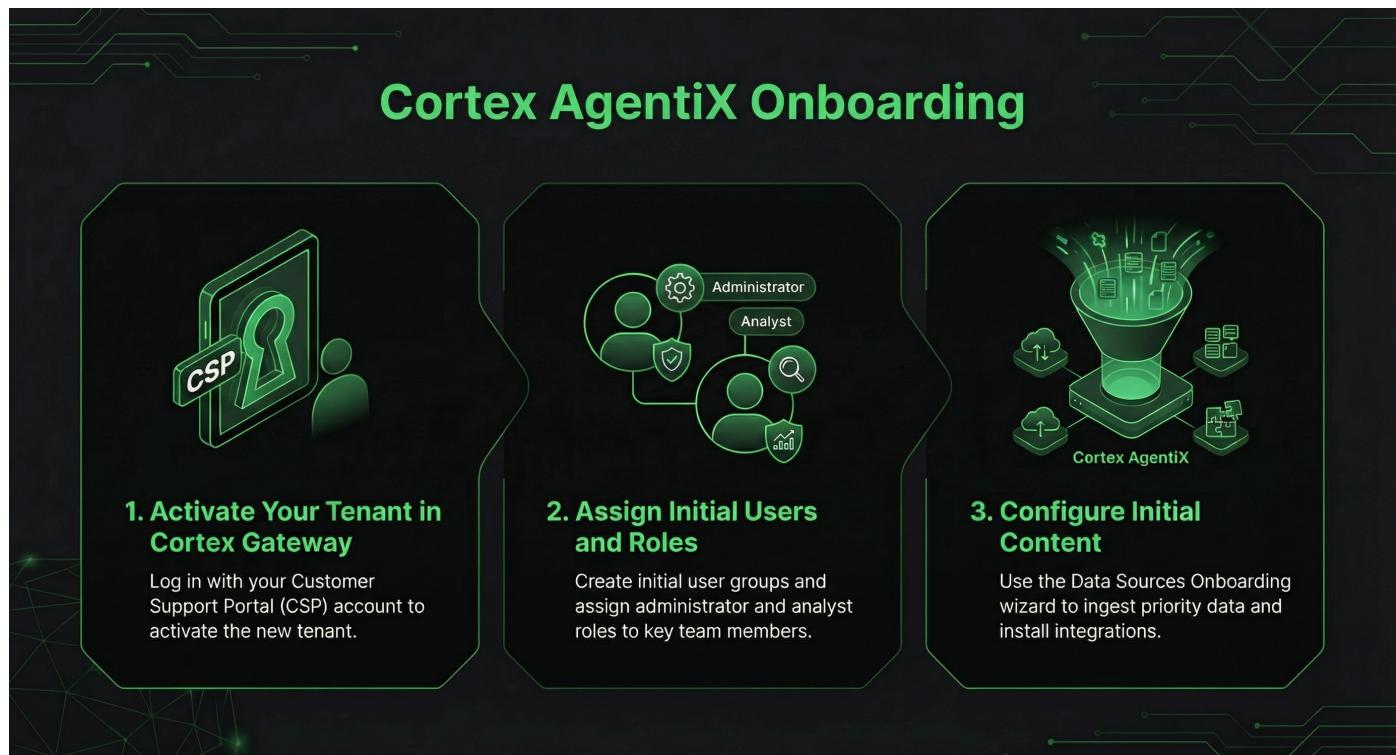


## 2.2 | Onboarding checklist

### Abstract

Activate, provision, grant access, and configure Cortex AgentiX.

Onboarding Cortex AgentiX involves establishing core environments first, such as activating a tenant, assigning user roles, and configuring agents.



Step	Details	See More
1. Activation and initial setup	<ul style="list-style-type: none"><li>Activate Cortex AgentiX in Cortex Gateway</li><li>Enable access to Palo Alto Networks resources</li></ul>	Activate Cortex AgentiX
2. Set up users and roles	Assign initial administrator and analyst user roles, create user groups, and assign roles to those groups (recommended) to a limited number of users initially. You can update this later.	Roles management
3: Configure content	<p>Use the Data Sources Onboarding wizard to configure the priority content for your use cases that you want to ingest into Cortex AgentiX.</p> <p>Configure/enable a key Threat Intelligence feed, such as the Unit 42 Intelligence feed, to enrich incoming issues. This ensures that as soon as a log/alert hits the Data Lake, it has the latest malicious context.</p>	Install and configure content

Your Cortex AgentiX tenant is now operational and is ingesting data.

## 2.3 | Activate Cortex AgentiX

### Abstract

Learn how to activate your tenant.

To activate a tenant, you need to log in to Cortex Gateway, a centralized portal for activating and managing tenants, users, roles, and user groups. After activating the tenant, you can then access the tenant. You must repeat this task for each tenant if you have multiple tenants. The activation process involves accessing Cortex Gateway, activating the tenant, and then accessing the tenant's resources.

### PREREQUISITE:



- The Cortex AgentiX activation email.
- A Customer Support Portal (CSP) account.

You need to set up your CSP account. For more information, see [How to Create Your CSP User Account](#).

When you create a CSP account, you can set up two-factor authentication (2FA) to log into the CSP by using an Email, Okta Verify, or Google Authenticator (non-FedRAMP accounts). For more information, see [How to Enable a Third Party IdP](#).

- You have one of the following roles assigned:

Role	Description
CSP role	The Super User role is assigned to your CSP account. The user who creates the CSP account is granted the Super User role.
Cortex role	<p>You must have the Account Admin role.</p> <p>If you are the first user to access Cortex Gateway with the CSP Super User role, you are automatically granted Account Admin permissions for the Cortex Gateway. You can also add Account Admin users as required.</p> <p>In the Cortex Gateway, you can activate new tenants, access existing tenants, and create and manage role-based access control (RBAC) for all of your tenants.</p>

#### How to activate Cortex AgentiX

1. Log in to Cortex Gateway.

You can also access the link from the activation email.

2. Enter your username and password or multi-factor authentication (if set up) by using your Customer Support Portal account credentials to sign in.

After you sign in, you can view the following:

- If you are a CSP Account Admin, you can see tenants allocated to your CSP account and ready for activation. After activation, you cannot move your tenant to a different CSP account.
- Tenant details such as license type, number of endpoints, and purchase date.
- Tenants that were activated and are now available. If you have more than one Customer Support Portal account, the tenants are displayed according to the Customer Support Portal account name.

3. In the Available for Activation section, use the serial number to locate the tenant that needs activation, and then click Activate.

4. On the Tenant Activation page, define the following:

Parameter	Description
Tenant Name	Enter the name of the tenant. Use a unique name across your company account up to 59 characters long.
Region	Geographic location where your tenant will be hosted. For more information about supported regions, see <a href="#">Cortex AgentiX supported regions</a> .
Tenant Subdomain	DNS record associated with your tenant. Enter a name that will be used to access the tenant directly using the full URL: <code>https://&lt;subdomain&gt;xdr.&lt;region&gt;.paloaltonetworks.com</code>

5. Review and agree to the terms and conditions of the Privacy policy, Terms of Use, and EULA , and then Activate your tenant.

#### NOTE:

Activation can take about an hour and does not require you to remain on the activation page. Cortex AgentiX sends a notification to your email when the process is complete.

6. After activation, from Cortex Gateway, in the Available Tenants, when hovering over the activated tenant, do the following:



- Ensure that you can successfully access the tenant by clicking the Cortex AgentiX tenant name (when the tenant is active).
- In the dialog box, view the tenant status, region, serial number, and license details.

**NOTE:**

If you want to change your tenant's name, the subdomain, or activate a development tenant (subject to license), on the right-hand side, click the ellipsis.

You can only change the subdomain once, and it cannot be undone.

After deleting the subdomain, you can reuse it after 7 days.

7. Enable access to Palo Alto Networks resources in your firewall. See [Enable access to Palo Alto Networks resources](#).

### 2.3.1 | Cortex AgentiX supported regions

#### Abstract

Supported regions in which you want to host Cortex AgentiX and any associated services.

The following table lists the regions available to host Cortex AgentiX and any associated Cortex services:

Country	Description
Australia (AU)	All Cortex AgentiX logs and ingested data remain hosted within the boundaries of Australia.
Canada (CA)	All Cortex AgentiX logs and ingested data remain hosted within the boundaries of Canada.
France (FR)	All Cortex AgentiX logs and ingested data remain hosted within the boundaries of France.
Germany (DE)	All Cortex AgentiX logs and ingested data remain hosted within the boundaries of Germany.
India (IN)	All Cortex AgentiX logs and ingested data remain hosted within the boundaries of India.
Japan (JP)	All Cortex AgentiX logs and ingested data remain hosted within the boundaries of Japan.
Netherlands (EU)	All Cortex AgentiX logs and ingested data remain hosted within the boundaries of the Netherlands.
Singapore (SG)	All Cortex AgentiX logs and ingested data remain hosted within the boundaries of Singapore.
South Korea (KR)	All Cortex AgentiX logs and ingested data remain hosted within the boundaries of South Korea.
United Kingdom (UK)	All Cortex AgentiX logs and ingested data remain hosted within the boundaries of the United Kingdom.
United States (US)	All Cortex AgentiX logs and ingested data remain hosted within the boundaries of the United States.

### 2.3.2 | Enable access to Palo Alto Networks resources

#### Abstract

Depending on your network environment settings, you may need to enable network access to the Cortex AgentiX resources.

After you receive your account details, in your firewall configuration, enable and verify access to Cortex AgentiX communication servers, storage buckets, and various resources.

**NOTE:**

Some of the IP addresses required for access are registered in the United States. As a result, some GeoIP databases do not correctly pinpoint the location from which IP addresses are used. All customer data is stored in your deployment region, regardless of the IP address registration, and data transmission is



restricted through any infrastructure to that region.

For IP address ranges in Google Cloud Platform (GCP), view the following tables for IP address coverage for your deployment:

- <https://www.gstatic.com/ipranges/goog.json>: IP address subnet ranges
- <https://www.gstatic.com/ipranges/cloud.json>: IP address ranges associated with your region

In your firewall configuration, enable the following resources:

FQDN	Description
<agentix-tenant>.xdr. <region>.paloaltonetworks.com	Used to connect to Palo Alto Networks. For the relevant region and IP address, see the <b>IP Address to connect to Cortex AgentiX</b> column below. For example, if the region is US, use the 35.244.250.18 IP address.
api-<agentix-tenant>.xdr. <region>.paloaltonetworks.com	Used for API requests and responses and to connect to an engine. For the relevant region and IP address, see the <b>API/EDL IP Address</b> column below. For example, if the region is US, use the 35.222.81.194 IP address.
ext-<agentix-tenant>.xdr. <region>.paloaltonetworks.com	Used for EDL (long-running integrations). For the relevant region and IP address, see the <b>API/EDL IP Address</b> column below. For example, if the region is US, use the 35.222.81.194 IP address.

#### NOTE:

<agentix-tenant> refers to the chosen subdomain of your Cortex AgentiX tenant, and <region> is the region in which your tenant is deployed.

The port is 443.

Egress is used for communication between Cortex AgentiX and customer resources. For the relevant region and IP address, see the **Egress IP Address** column below. For example, if the region is US, use 35.225.156.101 and 34.69.88.119.

Region	IP Address To Connect To Cortex AgentiX	API/EDL IP Address	Egress IP Address
CH (Switzerland)	34.111.6.153	34.65.248.119	<ul style="list-style-type: none"> <li>• 34.65.233.60</li> <li>• 34.65.222.25</li> </ul>
EU (Europe - Netherlands)	35.227.237.180	34.90.67.58	<ul style="list-style-type: none"> <li>• 34.147.67.188</li> <li>• 34.90.16.31</li> </ul>
US (United States)	35.244.250.18	35.222.81.194	<ul style="list-style-type: none"> <li>• 35.225.156.101</li> <li>• 34.69.88.119</li> </ul>

Outbound IPs for engines

Region	IP Addresses
US (United States)	<ul style="list-style-type: none"> <li>• 35.225.156.101</li> <li>• 34.69.88.119</li> </ul>
EU (Europe)	<ul style="list-style-type: none"> <li>• 34.147.67.188</li> <li>• 34.90.16.31</li> </ul>



Region	IP Addresses
CA (Canada)	<ul style="list-style-type: none"> <li>• 35.203.57.162</li> <li>• 35.203.90.79</li> </ul>
UK (United Kingdom)	<ul style="list-style-type: none"> <li>• 34.142.3.42</li> <li>• 34.142.44.136</li> </ul>
JP (Japan)	<ul style="list-style-type: none"> <li>• 34.146.60.215</li> <li>• 34.84.93.160</li> </ul>
SG (Singapore)	<ul style="list-style-type: none"> <li>• 35.240.144.192</li> <li>• 35.240.255.15</li> </ul>
AU (Australia)	<ul style="list-style-type: none"> <li>• 35.244.73.76</li> <li>• 35.201.22.63</li> </ul>
DE (Germany)	<ul style="list-style-type: none"> <li>• 34.107.83.197</li> <li>• 34.159.53.97</li> </ul>
IN (India)	<ul style="list-style-type: none"> <li>• 35.244.5.205</li> <li>• 34.93.118.113</li> </ul>
CH (Switzerland)	<ul style="list-style-type: none"> <li>• 34.65.222.25</li> <li>• 34.65.233.60</li> </ul>
PL (Poland)	<ul style="list-style-type: none"> <li>• 34.118.92.214</li> <li>• 34.116.223.119</li> </ul>
TW (Taiwan)	<ul style="list-style-type: none"> <li>• 104.199.223.229</li> <li>• 34.81.38.132</li> </ul>
QT (Qatar)	<ul style="list-style-type: none"> <li>• 34.18.39.0</li> <li>• 34.18.32.96</li> </ul>
FA (France)	<ul style="list-style-type: none"> <li>• 34.155.197.131</li> <li>• 34.155.5.100</li> </ul>
IL (Israel)	<ul style="list-style-type: none"> <li>• 34.165.46.47</li> <li>• 34.165.17.246</li> </ul>



Region	IP Addresses
SA (Saudi Arabia)	<ul style="list-style-type: none"> <li>• 34.166.58.243</li> <li>• 34.166.54.238</li> </ul>
ID (Indonesia)	<ul style="list-style-type: none"> <li>• 34.101.125.66</li> <li>• 34.101.218.184</li> </ul>
ES (Spain)	<ul style="list-style-type: none"> <li>• 34.175.255.99</li> <li>• 34.175.230.35</li> </ul>
IT (Italy)	<ul style="list-style-type: none"> <li>• 34.154.173.134</li> <li>• 34.154.229.60</li> </ul>
KR (South Korea)	<ul style="list-style-type: none"> <li>• 34.64.189.205</li> <li>• 34.64.45.118</li> </ul>
ZA (South Africa)	<ul style="list-style-type: none"> <li>• 34.35.70.193</li> <li>• 34.35.80.189</li> </ul>

In-app Help Center and notifications

FQDN	IP Addresses And Port
data.pendo.io	Port: 443
pendo-static-5664029141630976.storage.googleapis.com	Port: 443

Email notifications

IP address for all regions: 159.183.150.248

App login and authentication

FQDN	IP Address And Port
https://sso.paloaltonetworks.com	Port: 443

## 2.4 | Install and configure content

Abstract

Learn how content works in Cortex AgentiX.

Content is organized into content packs to support specific security orchestration use cases. Content packs can include multiple items such as actions, playbooks, scripts, dashboards, and widgets.

**NOTE:**



Cortex AgentiX includes AI system actions, used by system agents. Some system actions are available out-of-the-box with pre-installed content packs, such as Common Scripts and other system AI actions require you to download and install the relevant content pack.

To find and install additional content packs that include actions, go to Marketplace and select Content pack includes and Actions.

Content packs are either preinstalled or downloaded from one of the following:

- Data Sources

On the Data Sources & Integrations page, when you add a data source, the Data Onboarder wizard guides you through the steps for installing the integration instance and enables you to configure your integration. Cortex AgentiX automatically downloads and installs the required content packs for the integration, and recommends additional beneficial content, such as playbooks and dashboards relevant to the specific data source.

Use the Data Sources & Integrations page to configure content, which is designed for third-party integrations to help you onboard. For more information, see Add a new data source or instance.

- Marketplace

In Marketplace, you can download, install, manage, and contribute content packs. For example, the Microsoft Exchange Online content pack includes scripts, integrations, and playbooks. After downloading the content pack, configure the integration on the Data Sources & Integrations page to enable Cortex AgentiX communication with the Exchange server.

Content packs are created by Palo Alto Networks, technology partners, contributors, and customers. For more information about content packs, see Content packs.

**NOTE:**

Not all content packs are available to download on the Data Sources page. If you require content packs such as Phishing and Malware, you need to download the pack from Marketplace and configure the integration.

## 2.5 | Post-deployment steps

Abstract

Perform post-deployment tasks such as configuring your environment, pre-triggered automation rules, user roles, and log forwarding.

Perform post-deployment tasks such as setting up your environment, creating pre-triggered automation rules, and managing user roles and access management.

### 2.5.1 | Post-deployment checklist

Abstract

Use the post-deployment checklist to finish your Cortex AgentiX onboarding.

Start with the essential initial actions for post-deployment to get you up and running quickly. Continue with advanced actions, such as configuring relevant integrations and deploying an engine.

# Cortex AgentiX Post-Deployment

The diagram illustrates the post-deployment checklist with two main sections: INITIAL ACTIONS: GET UP AND RUNNING and ADVANCED ACTIONS: OPTIMIZE & EXTEND YOUR TENANT. A large green arrow points from left to right, indicating the progression of steps. The INITIAL ACTIONS section contains two cards: 'Configure Automations' (reviewing playbooks and scripts) and 'Validate Your Workflow' (investigating initial cases and issues). The ADVANCED ACTIONS section contains four cards: 'Set Up Integrations' (installing content packs), 'Manage Users and Roles' (configuring users and roles), 'Customize Your Environment' (adjusting server settings and configuring dashboards), and 'Deploy an Engine for On-Prem Access' (installing an engine for internal network automation).

INITIAL ACTIONS: GET UP AND RUNNING		ADVANCED ACTIONS: OPTIMIZE & EXTEND YOUR TENANT	
 <b>Configure Automations</b> Review playbooks and scripts, then apply automation rules to your use cases.	 <b>Validate Your Workflow</b> Investigate initial cases and issues to confirm security controls are functioning correctly.	 <b>Set Up Integrations</b> Install content packs from the Data Sources catalog or Marketplace for your tools.	 <b>Manage Users and Roles</b> Configure users, create user groups, and assign roles for access control.
 <b>Customize Your Environment</b> Adjust server settings, configure dashboards, and set up log forwarding.	 <b>Deploy an Engine for On-Prem Access</b> Install an engine to run automations on resources within your internal network.		



## Post-deployment - initial actions

The following table describes the post-deployment steps to get you up and running quickly.

Action	Details	See More
Configure automations	Review the different types of automations (playbooks and scripts) and apply automation rules to your use case.	Create an automation rule
Review cases and issues	Review the different types of automations (playbooks and scripts) and apply automation rules to your use case.  Review the different types of automations (playbooks and scripts) and apply automation rules to your use case.  Validate your workflow. Start with Widfire testing to confirm the security controls and sandbox integration are functional and working as expected.	Investigate cases  Investigate issues

## Post-deployment - Advanced

The following table describes the advanced post-deployment steps.

Action	Details	See More
Set up relevant integrations	Install essential content packs (for example, common security use cases or SOAR playbook) and configure relevant integrations, either from the Data Sources catalog or from Marketplace.  If your use case is not in the Data Sources catalog, you can install content from Marketplace. For example, if you require content packs such as Phishing and Malware, you need to download the pack from Marketplace and configure the integration. The Data Sources catalog includes the most used data sources to help you onboard.	Install and configure content
Update users and roles	Review and configure users, roles, and user groups as required. Each role extends specific privileges to users. The way you configure administrative access depends on your organization's security requirements. Use roles to assign specific access privileges to administrative user accounts.	Set up users and roles
Dashboards	Review and configure dashboards and ensure you can visualize activity from your active integrations.	About dashboards
Server and security settings	Customize and configure Cortex AgentX for a more personalized user experience: <ul style="list-style-type: none"><li>Server settings, such as the timezone, the timestamp format, password protection, and custom logos for communication task emails.</li><li>Security settings, such as allowed domains, allowed sessions, and user expiration.</li></ul>	Configure server settings  Configure security settings
Set up and deploy an engine	Deploy an engine if you have specific automation and feed integrations, scripts, or playbooks that execute actions or fetch data directly from resources within your internal network (for example, querying an on-prem Active Directory, isolating a machine via a local tool).  An engine is a proxy server application that is installed on a remote machine, enabling communication between the remote machine and the Cortex XSIAM tenant. You can run playbooks, scripts, commands, and integrations on the remote machine, and the results are returned to the tenant.	Set up an engine
Log forwarding	Set up sending logs to an external service, such as a Slack channel or an email distribution list.	Log forwarding



## 2.5.2 | Set up users and roles

### Abstract

Set up and configure roles and user groups in Cortex AgentiX and Cortex Gateway. Configure authentication and manage users.

Cortex uses role-based access control (RBAC) to manage roles with specific permissions for controlling user access. RBAC helps manage access to components, so that users, based on their roles, are granted the minimal access required to accomplish their tasks.

#### Task 1. Create user roles

Roles enable you to define permissions for specific components, such as issue data, playbooks, scripts, and jobs. For example, you can create a role that allows users to edit the properties of issues, but not delete issues. You can create new roles or customize out-of-the-box roles.

If you assign one or more roles to an issue, only users with those roles can view and interact with the issue. For example, you might have an issue with sensitive data that should only be accessible to Tier-1 analysts and managers.

Roles can also be used to define permissions for integration commands. On the Integration Permissions page, you can assign roles to specific integration instances (all commands for that instance) or specific integration instance commands. For example, you could assign the Generic Export Indicators Service integration instance the Account Admin role, or you could restrict certain commands in the Core Rest API to a specific role. For more information, see Integration Permissions.

For more information about out-of-the-box roles, permissions, and how to create roles, see Roles management.

#### Task 2. Create user groups

While roles can be assigned directly to users, we recommend creating user groups. Each user group has a single role associated with it, but each user group can contain multiple users, and user groups can be nested within each other, enabling you to further refine your RBAC requirements. Users can belong to several user groups.

For more information about user groups and how to create them, see User group management.

After adding users, assign those users to user groups or assign a direct role.

#### Task 3. Set up authentication

You can create users in the Customer Support Portal or by using SAML Single Sign-On (SSO) in the tenant. After you create users, they authenticate by doing the following:

- Authenticate through the Customer Support Portal
- Authenticate by using SAML Single Sign-On (SSO) in the Cortex tenant

For more information about setting up authentication, see Set up authentication.

#### Task 4. Manage users

By default, users do not have roles assigned and do not automatically have access to tenant data until you assign them a role or add them as members of a user group that has an assigned role.

For more information about how to manage users, see User management.

## 2.5.3 | Configure server settings

### Abstract

Configure server settings such as keyboard shortcuts, timezone, and timestamp format.

You can configure server settings such as keyboard shortcuts, timezone, timestamp format, and custom logos for communications task emails to create a more personalized user experience in Cortex AgentiX. Go to Settings → Configurations → General → Server Settings.

#### NOTE:

Keyboard shortcuts, timezone, and timestamp format are not set universally and only apply to the user who sets them.

Server Setting	Description
Keyboard Shortcuts	Enables you to change the default shortcut settings. The shortcut value must be a keyboard letter, A through Z, and cannot be the same for both shortcuts.



Server Setting	Description
Timezone	Select a specific timezone. The timezone affects the timestamps displayed in Cortex AgentiX, auditing logs and when exporting files.
Timestamp Format	<p>The format in which to display Cortex AgentiX data. The format affects the timestamps displayed in Cortex AgentiX, auditing logs and when exporting files.</p> <p>This setting is configured per user and not per tenant.</p>
Email Contacts	<p>A list of email addresses Cortex AgentiX can be used as a distribution list. The defined email addresses are used to send product maintenance, updates, and new version notifications. These addresses are in addition to the email addresses registered with your Customer Support Portal account.</p>
Custom Logo	<p>By default, the Cortex AgentiX logo displays on communication task emails. You can replace the default logo with a custom logo to match your organization's branding.</p> <p>Supported file formats are PNG, JPEG, SVG, and GIF.</p> <p>The minimum recommended image dimensions are 50px height and 50px width. The recommended maximum file size is 100 KB.</p>
AI Configuration	<ul style="list-style-type: none"> <li>Enable or disable the Cortex Agentic Assistant (Agents &amp; LLM Experience).</li> <li>Enable or disable AI case summarization capabilities.</li> </ul>
Password Protection (for downloaded files)	<p>Enable password protection when downloading retrieved files from an endpoint. This prevents users from opening potentially malicious files.</p> <p>Administrator permissions required.</p> <p><b>NOTE:</b></p> <p>If the Password Protection (for downloaded files) setting under Settings à Configuration à General à Server Settings is enabled, enter the password 'suspicious' to download the file.</p>
Google Maps Key	Enter the Google Maps API key to display the physical location of an entity on a Google map.



Server Setting	Description
Scope-Based Access Control (SBAC)	<p>Enforces granular scoping on users with a scoping configuration. A user can inherit scoping configurations from a user group, or have the scoping configuration applied directly on top of the role assigned from either a user group or a generated API Key.</p> <p>By default, Enable Scope Based Access Control is disabled and granular scoping is not enforced. Before enabling SBAC, we recommend that an administrator or a user with Access Management permissions first ensure that the users, user groups, and API Keys defined in Cortex AgentiX are granted the required access by assigning the relevant scopes. For more information, see <a href="#">Manage user scope</a>.</p> <p>(Optional) If enabled, you can select the Endpoint Scoping Mode, which is defined per tenant:</p> <ul style="list-style-type: none"> <li>• Permissive: Enables users with at least one scope tag to access the relevant entity with that same tag.</li> <li>• Restrictive: Users must have all the scoped tags that are tagged within the relevant entity of the system.</li> </ul>
Data Ingestion Monitoring (Beta)	<p>Data ingestion health monitors the availability and overall health of data collection. When enabled, Cortex AgentiX creates the following types of alerts:</p> <ul style="list-style-type: none"> <li>• <b>Ingestion health alerts:</b> Based on the data ingestion metrics and indicate disruptions in data collection</li> <li>• <b>Collection health alerts:</b> Based on error statuses in collection integrations and indicate that a collector is not connected</li> </ul> <p>If you disable data ingestion monitoring, Cortex AgentiX continues to collect metrics, but alerts are not created.</p> <p>Related information</p> <ul style="list-style-type: none"> <li>• Use data ingestion health metrics in Cortex Query Language queries and to create correlation rules with your data ingestion logic. For more information, see <a href="#">Monitor data ingestion health</a>.</li> <li>• View all health alerts on the <a href="#">Health Alerts</a> page. For more information, see <a href="#">About health issues</a>.</li> </ul>
XQL Configuration	<p>Enables setting case sensitivity across Cortex AgentiX.</p> <p>By default, this setting is set to <code>false</code> and field values are evaluated as case insensitive.</p>
Define the cases target MTTR per issue severity	<p>Determines within how many days and hours you want issues resolved according to the issue severity Critical, High, Medium, and Low.</p> <p>The defined MTTR is used to display the Resolved Issue MTTR dashboard widgets.</p>



Server Setting	Description
Impersonation Role	<p>The type of role permissions granted to the Palo Alto Networks Support team when opening support tickets. We recommend that role permissions be granted only for a specific time frame, and full administrative permissions be granted only when specifically requested by the Support team.</p> <p>Role permissions include:</p> <ul style="list-style-type: none"> <li>• Read-only: Default setting; grants read-only access to your tenant.</li> <li>• Support-related actions: Grants permissions to tech support file collection, dump file collection, investigation query, correlation rule, BIOC and IOC rule editing, alert starring, exclusion, and exception editing</li> <li>• Full role permissions: No limitations are applied; grants full permissions to all actions and content on your tenant</li> </ul> <p>Permission Reset Timeframe: Determines how long role permissions are valid.</p>
Custom Content	<ul style="list-style-type: none"> <li>• <b>Export all custom content:</b> Exports custom content, such as playbooks and scripts as a content bundle, which you can import to another Cortex AgentIX tenant.</li> <li>• <b>Upload custom content:</b> Imports custom content created from another Cortex AgentIX tenant.</li> </ul>
Case display modes	Allow users the access the Cases page in legacy mode.
Caching	Improve performance on the Cases and Issues pages by enabling a temporary data cache.
Issues	Create timer fields that display in the issues table and issue layouts. For more information, see Configure issue timer fields.
Indicators	<p><b>NOTE:</b></p> <p>Requires the TIM add-on.</p> <p>If you have the TIM add-on, you can enable or disable system-wide automatic indicator extraction and enrichment from issues.</p>

#### 2.5.4 | Configure security settings

##### Abstract

Configure security settings such as session expiration, user login expiration, and dashboard expiration.

You can configure security settings such as how long users can be logged in Cortex AgentIX, and from which domains and IP ranges users can log in.



Settings	Options	Description
Session Expiration	User Login Expiration	The number of hours (between 1 and 24) after which the user's login session expires. You can also choose to automatically log users out after a specified period of inactivity.
	Dashboard Expiration	Whether the Dashboard page expires at the same time as the user login session or after seven days. This is useful when you view a dashboard on a separate screen.  For example, if you select seven days for dashboards and eight hours for login expiration, and you are currently viewing the Dashboard page, the dashboard expiration takes priority (seven days). This ensures that the Dashboard page continues to display the widgets for an extended period.
Allowed Sessions	Approved Domains	The domains from which you want to allow user access (login) to Cortex AgentiX. You can add or remove domains as necessary.
	Approved IP Ranges	The IP ranges from which you want to allow user access (login) to Cortex AgentiX. You can also choose to limit API access from specific IP addresses.
User Expiration	Deactivate Inactive User	Deactivate an inactive user, and also set the user deactivation trigger period. By default, user expiration is disabled. When enabled, enter the number of days after which inactive users should be deactivated.
Same-Site Cookie Policy	Strict	Configure your Cortex tenant's SameSite cookie security policy by selecting between two settings to control how users log in from external links: <ul style="list-style-type: none"> <li>• Strict (Recommended): Requires users to reauthenticate when clicking a link from another site, even if they are already signed in.</li> </ul>
	Lax	<ul style="list-style-type: none"> <li>• Lax: Offers a more seamless experience by allowing users to access the tenant directly from external links without needing to log in again. Yet, we advise against this setting for security reasons.</li> </ul>
Allowed Domains	Domain Name	The domain names that can be used in your distribution lists for reports. For example, when generating a report, ensure the reports are not sent to email addresses outside your organization.



## 2.5.5 | Set up an engine

### Abstract

Set up a Cortex AgentiX engine on a remote machine.

Engines are installed on a remote machine and used mainly for the following:

- Integration instances for on-prem applications. For example, the GitLab v2 integration enables you to run commands on GitLab instances.
- Execute scripts and commands that require access to on-prem resources. For example, the Active Directory v2 integration enables you to run commands in Active Directory.
- Generic Indicator export service. In Cortex AgentiX, you can configure an EDL to share a list of Cortex AgentiX indicators with other products in your network, such as a firewall or SIEM. For example, your Palo Alto Networks firewall can add IP addresses and domain data from the EDL to block or allow lists.
- Load balancing enables the distribution of the command execution load.

Before installation, we recommend you review the engine requirements for hardware and operating systems. Engines can be installed on Linux machines running a variety of operating systems, including Ubuntu, RHEL, Oracle Linux, and Amazon Linux.

1. Select Settings → Configurations → Data Broker → Engines → Create New Engine.
2. In the Engine Name field, add a meaningful name for the engine.
3. Select one of the installer types from the list.
4. (Optional) (Shell only) Select the checkbox to enable multiple engines to run on the same machine.  
If you have an existing engine, and you did not select the checkbox, and now you want to install another engine on the same machine, you need to delete the existing engine.
5. (Optional) Add any required configuration in JSON format.
6. Click OK to create the engine.

To learn more about engines, requirements, and installation, see [Install an engine](#).

## 2.5.6 | Log forwarding

### Abstract

Stay informed and updated about events in your system by forwarding alerts and reports to an external service, such as a syslog receiver, a Slack channel, or an email account.

Logs provide information about events that occur in the system. These logs are a valuable tool in troubleshooting issues that might arise in your Cortex AgentiX tenant.

To stay informed about important alerts and events, you can configure your notifications and specify the type of logs you want to forward. You can choose to receive these notifications through an email account, a Slack channel, or a syslog receiver.

### 2.5.6.1 | Forward logs and data from Cortex AgentiX to external services

#### Abstract

Learn how to forward logs and data from Cortex AgentiX to external third-party services such as email, Slack, syslog, and Splunk.

You can forward logs, cases, and issues from Cortex AgentiX to an external service. By forwarding logs and data, you can manage alerts and investigations in external systems and meet data retention requirements. Available services include the following:

- **Slack channel and/or syslog receiver:** Configure the external application with Cortex AgentiX. After the application is configured, configure notification forwarding, specifying the data/log type you want to forward.
- **Email distribution list:** Configure notification forwarding, specifying the data/log type you want to forward.
- **Splunk, Amazon SQS, Amazon S3, and Webhook:** Only cases and issues can be forwarded to these services. The external application must be configured in Cortex AgentiX and egress configured in the Cortex Gateway before forwarding to these services.

The following table shows the log types supported for each notification type:



Data/Log Type	Email	Slack	Syslog	Splunk, Amazon SQS, Amazon S3, Webhook
Issues	â	â	â	â
Cases	â	â	â€	â
Management Audit Logs	â	â€	â	â€
Health Issues (Deprecated)	â	â	â	â€

#### 2.5.6.1.1 | Configure external applications for forwarding

##### Abstract

Configure external applications so you can forward data to services such as syslog servers, Slack, Splunk, Amazon SQS, Amazon S3, and Webhook.

Data and logs can be forwarded to third-party external services. The external service must be configured in Cortex AgentiX before you set up notification forwarding.

Before forwarding cases or issues to Splunk, Amazon S3, Amazon SQS, or Webhook, you need to configure egress in the Cortex Gateway. You do not need to configure egress for email, Slack, or syslog forwarding.

##### NOTE:

- No prior configuration is required to send data or logs to an email distribution list.
- Only cases and issues can be forwarded to Slack, Amazon S3, Amazon SQS, Splunk, and Webhook.
- To forward data to Amazon SQS, the following permissions are required: `sqs:GetQueueAttributes`, `sqs>ListQueues`, `sqs:SendMessage`, `sqs:SendMessageBatch`, `tag:GetResources`, and `iam:GetRole`.
- To forward data to Amazon S3, the following permissions are required: `s3:PutMessage`, `s3:PutObject`, and `s3>ListBucket`.

##### Task 1: Configure egress in Cortex Gateway

To forward data to Amazon S3, Amazon SQS, Splunk, or Webhook, you must first configure egress in Cortex Gateway. You do not need to configure egress to forward to Slack, email, or a syslog server.

##### NOTE:

Only a user with Account Admin or Instance Admin permissions can configure egress. For more information, see [Egress configurations](#).

1. In the Cortex Gateway, go to Permission Management â Egress Configurations â Path.
2. Select the account name and tenant.
3. In the Flow field, select the third-party service you want to forward to.

##### NOTE:

For Amazon SQS or Amazon S3, you need to provide the queue/bucket name. For Webhook or Splunk, you need to provide the domain, including any subdomain.

4. Add the configuration.

##### Task 2: Configure access in your firewall

If you are forwarding to Splunk or Webhook, add the IP addresses for your tenant region to your firewall. For more information, refer to the list of ingress IPs in [Enable access to required PANW resources](#).

##### Task 3: Configure external application

##### NOTE:

You can also configure external applications when you create a new forwarding configuration. After defining the configuration and setting the scope, you can [Add Application](#) and follow the instructions below for any of these destinations.

1. Go to Settings â Configurations â Integrations â External Applications â Add Application.

2. Choose one of the following applications:

Amazon S3

1. Enter the S3 URI.



2. Click Verify.

**NOTE:**

If egress has not been configured in the Cortex Gateway, verification will fail and a message will display that the endpoint does not match any approved routes.

3. After verification is successful, you can enter the instance name and optional description.

4. Enter the Role ARN (Amazon Resource Name).

5. Enter the AWS region. For example, `eu-central-1`.

6. (Optional) Select the maximum duration to collect data before writing to a new file. The default is one hour.

7. Test the connection.

#### Amazon SQS

1. Enter the Queue URL.

2. Click Verify.

**NOTE:**

If egress has not been configured in the Cortex Gateway, verification will fail and a message will display that the endpoint does not match any approved routes.

3. After verification is successful, you can enter the name and optional description.

4. Select either IAM Role or IAM Access Keys.

- For IAM Role, enter the Role ARN (Amazon Resource Name).
- For IAM Access Keys, enter the Access Key and Secret Key.

5. Test the connection.

#### Slack

For Slack, see Integrate Slack for outbound notifications.

#### Syslog

For syslog server configuration, see Integrate a syslog receiver.

#### Splunk

1. Enter the Splunk HTTP event collector URL. The URL can include a port. The connection must be HTTPS.

2. Click Verify.

**NOTE:**

If egress has not been configured in the Cortex Gateway, verification will fail and a message will display that the endpoint does not match any approved routes.

3. After verification is successful, you can enter the instance name and optional description.

4. Enter the authentication token for secure access to your Splunk instance.

5. Test the connection.

#### Webhook

1. Enter the Webhook URL. The URL can include a port. The connection must be HTTPS.

2. Click Verify.

**NOTE:**

If egress has not been configured in the Cortex Gateway, verification will fail and a message will display that the endpoint does not match any approved routes.

3. After verification is successful, you can enter the instance name and optional description.

4. (Optional) Show advanced settings to add HTTP headers.

5. Test the connection.

3. Click Connect.

#### Task 4: Configure notification forwarding

Configure notification forwarding to email or external services. For more information, see [Configure notification forwarding](#).



## Abstract

Define syslog settings and then configure notification forwarding to receive notifications about issues and reports.

A syslog receiver can be a physical or virtual server, a SaaS solution, or any service that accepts syslog messages.

To send Cortex AgentX notifications to your syslog receiver, you first need to define the settings for the syslog receiver. After this is complete, you can configure notification forwarding.

## Enable access

Before you begin, enable access to the following Cortex AgentX IP addresses for your region in your firewall.

### Americas

Region	Log Forwarding Address
United States - Americas (US)	35.232.87.9, 35.224.66.220
United States - Government	104.198.222.185, 35.239.59.210
Brazil (BR)	35.247.234.13, 34.39.178.116
Canada (CA)	35.203.54.204, 35.203.52.255

### EMEA (Europe, Middle East, Africa)

Region	Log Forwarding IP Addresses
France (FA)	34.163.100.253, 34.155.72.149
Germany (DE)	35.234.95.96, 35.246.192.146
Israel (IL)	34.165.194.4, 34.165.101.105
Italy (IT)	34.154.0.173, 34.154.71.94
Netherlands - Europe (EU)	34.90.202.186, 34.90.105.250
Poland (PL)	34.118.45.145, 34.118.126.170
Qatar (QT)	34.18.48.182, 34.18.43.40
Saudi Arabia (SA)	34.166.50.215, 34.166.55.72
South Africa (ZA)	34.35.70.253, 34.35.10.167
Spain (ES)	34.175.83.90, 34.175.230.150



Region	Log Forwarding IP Addresses
Switzerland (CH)	34.65.228.95, 34.65.74.83
United Kingdom (UK)	34.105.227.105, 34.105.149.197

JPAC (Asia-Pacific)

Region	Log Forwarding IP Addresses
Australia (AU)	35.189.38.167, 34.87.219.39
Delhi (DL)	34.126.223.198, 34.131.110.15
India (IN)	34.93.247.41, 34.93.183.131
Indonesia (ID)	34.101.248.99, 34.101.176.232
Japan (JP)	34.84.88.183, 35.243.76.189
Singapore (SG)	35.240.192.37, 34.87.125.227
South Korea (KR)	34.64.198.58, 34.47.86.20
Taiwan (TW)	35.234.2.208, 35.185.171.91

How to send issues or logs to a syslog receiver

1. Go to Settings → Configurations → Integrations → External Applications → Add Application and select Syslog.
2. Define the following parameters:

Parameter	Description
Name	Unique name for the server profile.
Destination	IP address or fully qualified domain name (FQDN) of the syslog receiver.
Port	Port number to send syslog messages.
Facility	Select one of the syslog standard values. The value maps to how your syslog server uses the facility field to manage messages. For details on the facility field, see RFC 5424.



Parameter	Description
Protocol	<p>Method of communication with the syslog receiver:</p> <ul style="list-style-type: none"> <li>• TCP: No validation is made on the connection with the syslog receiver. However, if an error occurred with the domain used to make the connection, the Test connection will fail.</li> <li>• UDP: No error checking, error correction, or acknowledgment. No validation is done for the connection or when sending data.</li> <li>• TCP + SSL: Cortex AgentiX validates the syslog receiver certificate and uses the certificate signature and public key to encrypt the data sent over the connection.</li> </ul>
Certificate	<p>The communication between Cortex AgentiX and the syslog destination can use TLS. In this case, upon connection, Cortex AgentiX validates that the syslog receiver has a certificate signed by either a trusted root CA or a self-signed certificate. You may need to merge the Root and Intermediate certificate if you receive a certificate error when using a public certificate.</p> <p>If your syslog receiver uses a self-signed CA, upload your self-signed syslog receiver CA. If you only use a trusted root CA leave the certificate field empty.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>• Up to TLS 1.3 is supported.</li> <li>• Verify the self-signed CA includes your public key.</li> </ul> <p>You can ignore certificate errors. For security reasons, this is not recommended. If you choose this option, data and logs will be forwarded even if the certificate contains errors.</p>

3. Test the parameters to ensure a valid connection, and click Connect when ready.

You can define up to five syslog receivers. Upon success, the table displays the syslog servers and their status.

After you integrate with your syslog receiver, configure your forwarding settings. For more information see, [Configure notification forwarding](#).

#### Syslog receiver test message errors

When configuring a syslog message, Cortex AgentiX sends a test message. If a test message cannot be sent, Cortex AgentiX displays an error message to help you troubleshoot.

The following table includes descriptions and suggested solutions for the error messages:

Error Message	Description
Host Resolving Failed	The IP address or hostname you provided doesn't exist, or can't be resolved.
Configured Local Address	The IP address or hostname you provided is internal and can't be used.



Error Message	Description	
Wrong Certificate Format	<p>The certificate you uploaded is in an unexpected format and can't be used. The certificate must be an ASCII string or a bytes-like object.</p>	<p>Re-create the certificate in the correct format, for example:</p> <pre data-bbox="469 281 699 348">-----BEGIN CERTIFICATE----- MIIDHTCCAgNgAwIBAgIQSwieRyGdh6BNRQyp406bnTANBgkqhkiG9w0BAQsFADAhMR8wHQYDVQQDExZTVVJTLUNoYXJsaWVBbHBoYS1Sb290MB4XDTHwMDQzMDE4MjEzNFO -----END CERTIFICATE-----</pre>
Connection Timed Out	<p>Cortex AgentiX didn't connect to the syslog receiver in the expected time. This could be because your firewall blocked the connection or because the configuration of the syslog server caused it to drop the connection.</p>	<p>Check the firewall logs and the connection using Wireshark.</p>
Connection Refused	<p>The syslog receiver refused the connection. This could be because your firewall blocked the connection or because the configuration of the syslog server caused it to drop the connection.</p>	<p>Check the firewall logs and the connection using Wireshark.</p>
Connection Reset	<p>The connection was reset by the syslog receiver. This could be because your firewall blocked the connection or because the configuration of the syslog receiver caused it to drop the connection.</p>	<p>Check the firewall logs and the connection using Wireshark.</p>



Error Message	Description	
Certificate Verification Failed	<p>The uploaded certificate couldn't be verified for one of the following reasons.</p> <ul style="list-style-type: none"> <li>The certificate doesn't correspond to the certificate on the syslog receiver and cannot be validated.</li> <li>The certificate doesn't have the correct hostname.</li> <li>You are using a certificate chain and didn't merge the certificates into one certificate.</li> </ul>	<ul style="list-style-type: none"> <li>Incorrect certificate: to check that the certificate you are uploading corresponds to the server syslog certificate, use the following OpenSSL command:  <pre>openssl verify -verbose -CAfile cortex_upload_certificate syslog_certificate</pre> If the certificate is correct, the result is <b>syslog_certificate: OK</b>.</li> <li>Incorrect hostname: make sure that the hostname/ip in the certificate matches the syslog server.</li> <li>Certificate chain: If you are using a list of certificates, merge the chain into one certificate. You can concatenate the certificates:  <pre>cat intermediate_cert root_cert &gt; merged_syslog.crt</pre> If the concatenated certificate doesn't work, change the order of the root and intermediate certificates, and try again.</li> </ul> <p>To verify that the chain certificate was saved correctly, use the following OpenSSL command.</p> <pre>openssl verify -verbose -CAfile cortex_upload_certificate syslog_certificate</pre> <p>If the certificate is correct, the result is <b>syslog_certificate: OK</b>.</p>
Connection Terminated Abruptly	The firewall or the syslog receiver dropped the connection unexpectedly. This could be because the firewall on the customer side limits the number of connections, the configuration on the syslog receiver drops the connection, or the network is unstable.	Check the firewall logs and the connection using Wireshark.
Host Unreachable	The network configuration is faulty, and the connection can't reach the syslog receiver.	Check the network configuration to make sure everything is configured correctly, like a firewall or a load balancer which may be blocking the connection.



Error Message	Description
SSL Error	Unknown SSL error. To investigate the issue, contact support.
Connection Unavailable	General error. To investigate the issue, contact support.

#### 2.5.6.1.1.2 | Integrate Slack for outbound notifications

##### Abstract

Learn how to integrate Cortex AgentiX with your Slack workspace and stay updated on important alerts and events.

Integrate Cortex AgentiX with your Slack workspace to manage and highlight your issues and reports. Creating a Cortex AgentiX Slack channel ensures that defined issues are exposed on laptop and mobile devices using the Slack interface. Unlike email notifications, Slack channels provide dedicated spaces where you can contact specific members regarding your issues.

##### How to integrate Slack with Cortex AgentiX

1. Go to Settings → Configurations → Integrations → External Applications → Add Application and click Slack.

2. Click Ok to go to an external Slack page to install Cortex AgentiX on your Slack workspace.

##### NOTE:

You are directed to the Slack browser to install Cortex AgentiX. You can only use this link to install Cortex AgentiX on Slack. Attempting to install from Slack Marketplace will redirect you to Cortex AgentiX documentation.

3. Click Submit.

Upon successful installation, Cortex AgentiX displays the workspace to which you connected.

##### What to do next

After you integrate with your Slack workspace, configure your forwarding settings. For more information see, Configure notification forwarding. To send reports to Slack, see Run or schedule reports.

#### 2.5.6.1.2 | Configure notification forwarding

##### Abstract

Learn how to create a forwarding configuration that specifies the log type you want to forward.

After you integrate with an external service such as Slack, a syslog server, Amazon S3, Amazon SQS, Webhook, or Splunk, create a forwarding configuration that specifies the data or log type you want to forward. You can configure notifications for issues, cases, and logs. To send reports to email or Slack, see Run or schedule reports.

##### PREREQUISITE:

Before you can select an external service for notification forwarding, you must integrate the external service with Cortex AgentiX. For more information, see Configure external applications for forwarding. No prior configuration is required to send data to an email distribution list.

##### How to configure notifications

1. Select Settings → Configurations → General → Notifications → Add Forwarding Configuration.

2. Enter a name for the configuration.

3. Select the data or log type you want to forward:



- Issues: Send notifications for specific issue types.

**NOTE:**

- **Forwarding destinations:** Only issues and cases can be forwarded to Slack, Splunk, Amazon SQS, Amazon S3, or Webhook.
  - **Notification forwarding by domain:** To configure notification forwarding for issues by domain, select Issues and filter the Issues table by Issue Domain.
  - **Case IDs in notifications:** If case matching completes before the notification is sent, the Case ID is included. If matching takes longer than the notification trigger, the notification is sent without a Case ID to prioritize timely visibility in the target system.
  - **Alert vs. issue format:** By default, new configurations use the issue format, but you can select the alert format if needed, when forwarding to email, Slack, or a syslog server. You cannot forward issues in the alert format to Splunk, Amazon SQS, Amazon S3, or Webhook.
- Existing legacy configurations are not automatically updated and continue to send notifications in the alert format. To use the issue format, edit the existing configuration.

- Management Audit Logs: Send notifications for audit logs about events related to your Cortex AgentiX tenant.
- Cases – Send notifications for specific cases.

**NOTE:**

Not all data and log types can be sent to all external services. For more information, see Forward logs and data from Cortex AgentiX to external services.

4. (Optional) Enter a description of the forwarding configuration.

5. Click **Next**, and under **Scope**, filter which issues, cases, or logs you want included in a notification.

For example, for a filter set to **Severity = Medium, Category = Configuration**, Cortex AgentiX sends the issues or events matching this filter as a notification.

6. Click **Next**.

7. Select email or the external service you want to forward to.

#### Email (Issues, cases, logs)

1. Enable the email option and click Email to expand the form.
2. Enter the email address for your Distribution List.
3. For issue forwarding, you can define the Grouping Timeframe, which is the time frame, in minutes, to specify how often Cortex AgentiX sends notifications. Every 20 issues aggregated within this time frame are sent together in one notification, sorted according to severity. To send a notification when one issue is generated, set the time frame to 0. The grouping time frame for case and management audit log is 10 minutes and cannot be modified.
4. (Optional) Define your email configuration:
  - a. In the Distribution List, add the email addresses to which you want to send email notifications.
  - b. Choose whether you want Cortex AgentiX to provide an auto-generated subject.
- 5.
6. Choose whether you want Cortex AgentiX to provide an auto-generated subject or enter your own subject.

**NOTE:**

The Grouping Timeframe defines the time frame, in minutes, of how often Cortex AgentiX sends notifications. Every 20 issues or 20 events aggregated within this time frame are sent together in one notification, sorted according to severity. To send a notification when one issue or event is generated, set the time frame to 0.

#### Syslog server (Issues, logs)

1. Enable the Syslog option and click Syslog to expand the form.
2. Select a syslog receiver. Cortex AgentiX displays the list of receivers integrated with your Cortex AgentiX tenant.

#### Slack (Issues, cases)

1. Enable the Slack option and click Slack to expand the form.
2. Enter the Slack channel name and select from the list of available channels. Slack channels are managed independently of Cortex AgentiX in your Slack workspace. After integrating your Slack account with your Cortex AgentiX tenant, Cortex AgentiX displays a list of specific Slack channels associated with the integrated Slack workspace.

#### Amazon S3, Amazon SQS, Splunk, or Webhook (Issues, cases)

1. Enable the Amazon S3, Amazon SQS, Splunk, or Webhook option and click to expand the form.



2. Select the instance name.

8. Click Next.

9. Review the forwarding configuration and click Create.

#### 2.5.6.1.3 | Monitor administrative activity

##### Abstract

View all Cortex AgentX administrator-initiated actions taken on issues, cases, and live terminal sessions.

From Settings â†’ Management Audit Logs, you can track the status of all administrative and investigative actions. Cortex AgentX stores audit logs for 365 days. Use the page filters to narrow the results or manage tables to add or remove fields as needed.

To ensure you and your colleagues stay informed about administrative activity, you can configure notification forwarding to forward your Management Audit log to an email distribution list, Syslog server, or Slack channel.

The following table describes the default **and optional fields** that you can view in alphabetical order.

Field	Description
Email	Email address of the administrative user
Description	Descriptive summary of the administrative action. Hover over this field to view more detailed information in a pop-up tooltip. This enables you to know exactly what has changed, and, if necessary, roll back the change.
Host Name	Name of any relevant affected hosts
ID	Unique ID of the action
Result	Result of the administrative action: Success, Partial, or Fail.
Subtype	Subcategory of action
Timestamp	Time and date of the action



Field	Description
Type	<p>Type of activity logged, one of the following:</p> <ul style="list-style-type: none"> <li>• Issue Exclusions: Suppression of particular issues from Cortex AgentiX .</li> <li>• Issue Fields: Modification of issue fields.</li> <li>• Issue Layouts: Modification of issue layouts.</li> <li>• Issue Layout Rules: Modification of issue layout rules.</li> <li>• Issue Notifications: Modification of the format or timing of issues.</li> <li>• Issue Rules: Modification of issue rules.</li> <li>• API Key: Modification of the Cortex AgentiX API key.</li> <li>• Authentication: User sessions started, along with the user name that started the session.</li> <li>• Dashboards: Use of particular dashboards.</li> <li>• Case Management: Actions taken on cases, issues, and artifacts in cases.</li> <li>• Ingest Data: Import of data for immediate use or storage in a database.</li> <li>• Integrations: Integration operations, such as integrating Slack for outbound notifications.</li> <li>• Licensing: Any licensing-related operation.</li> <li>• Managed Threat Hunting: Activity relating to managed threat hunting.</li> <li>• MSSP: Management of security services providers.</li> <li>• Public API: Authentication activity using an associated Cortex AgentiX API key.</li> <li>• Query Center: Operations in the Query Center.</li> <li>• Remediation: Remediation operations.</li> <li>• Reporting: Any reporting activity.</li> <li>• Response: Remedial actions taken. For example: Isolate a host, undo host isolation, add a file hash signature to the block list, or undo the addition to the block list.</li> <li>• Rules: Modification of rules.</li> <li>• Rules Exceptions: Creation, editing, or deletion under Rules exceptions.</li> <li>• SaaS Collection: Any collected SaaS data.</li> <li>• Script Execution: Any script execution.</li> <li>• Starred Cases: Modification of starred cases.</li> <li>• Vulnerability Assessment: Any vulnerability assessment activity.</li> </ul>
User Name	The user who performed the action.

#### 2.5.6.2 | Data and log notification formats

##### Abstract

Cortex AgentiX provides you with different formats for its log notifications.

When Cortex AgentiX cases, issues, and logs are forwarded to email or a third-party system, notifications are sent in a specific format.

**NOTE:**



Issues can be forwarded to email, syslog servers, and Slack in the alert format, if you prefer. The alert format can be selected when you configure your forwarding notification.

#### 2.5.6.2.1 | Issue notification format

##### Abstract

Learn about the formats used to forward issues to third-party services.

Issues can be forwarded to the following:

- Email distribution list
- Syslog server
- Slack
- Splunk, Amazon SQS, Amazon S3, or Webhook

##### Email account

Cortex AgentIX sends issues to email accounts based on the settings you configure. Email messages also include an issue code snippet of the fields according to the columns in the Issue table.

The notification format is as follows:

- If only one issue exists in the queue, a single-issue email format is sent.
- If more than one issue was grouped in the time frame, all the issues in the queue are forwarded together in a grouped email format.

Example 1.

Single-issue email message

```
Email Subject: Issue: <issue_name>
Email Body:
  Issue Name: Suspicious Process Creation
  Severity: High
  Source: Correlation
  Category: Malware
  Action: Detected
  Host: <host name>
  Username:<user name>
  Excluded: No
  Starred: Yes
  Issue: <link to the tenant issue view>
  Case: <link to the tenant case view>
```

Example 2.

Grouped issue email message

```
Email Subject: Issues: <first_highest_severity_issue> + x others
Email Body:
  Issue Name: Suspicious Process Creation
  Severity: High
  Source: Correlation
  Category: MalwareAction: Detected
  Host: <host name>
  Username:<user name>
  Excluded:No
  Starred: Yes
  Issue: <link to the tenant issue view>
  Case: <link to the tenant case view>
  Issue Name: Behavioral Threat Protection
  Issue ID: 2412
  Description: A really cool detection
  Severity: Medium
  Source: Correlation
  Category: Exploit
  Action: Prevented
  Host: <host name>
  Starred: Yes
  Case: <link to the tenant issue view>
  Issue: <link to the tenant case view>
  Notification Name: ª My notification policy 2 ª
  Notification Description: ª Starred issues with medium severity ª
```

Example 3.

Email body

```
{
  "original_issue_json":{
```



```

"recordType": "threat",
"customerId": "<Customer ID>",
"severity": 4,
"....",
"is_pcap": null,
"contains_featured_host": [
    "NO"
],
"contains_featured_user": [
    "YES"
],
"contains_featured_ip": [
    "YES"
],
"events_length": 1,
"is_excluded": false
}

```

Slack channel, Splunk, Amazon S3, Amazon SQS, Webhook

You can send issue notifications to a single Slack contact or a Slack channel, or to Splunk, Amazon S3, Amazon SQS, or Webhook. Notifications are similar to the email format.

Syslog receiver

Issue notifications forwarded to a syslog receiver are sent in a CEF format RF 5425.

Section	Description
Syslog header	<9>: PRI (considered a priority field)1: version number2020-03-22T07:55:07.964311Z: timestamp of when alert/log was sentcortexxdr: host name
CEF header	HEADER/Vendor="Palo Alto Networks" (as a constant string)HEADER/Device Product="Cortex XDR" (as a constant string)HEADER/Product Version= Cortex XDR version (2.0/2.1.....)HEADER/Severity=(integer/0 - Unknown, 6 - Low, 8 - Medium, 9 - High)HEADER/Device Event Class ID=alert sourceHEADER/name =alert name
CEF body	end=timestamp shost=endpoint_name deviceFacility=facility cat=category externalId=external_id request=request cs1=initiated_by_process cs1Label=Initiated by (constant string) cs2=initiator_commande cs2Label=Initiator CMD (constant string) cs3=signature cs3Label=Signature (constant string) cs4=cgo_name cs4Label=CGO name (constant string) cs5=cgo_command cs5Label=CGO CMD (constant string) cs6=cgo_signature cs6Label=CGO Signature (constant string) dst=destination_ip dpt=destination_port src=source_ip spt=source_port fileHash=file_hash filePath=file_path targetprocesssignature=target_process_signature tenantname=tenant_name tenantCDLid=tenant_id CSPaccountname=account_name initiatorSha256=initiator_hash initiatorPath=initiator_path osParentName=parent_name osParentCmd=parent_command osParentSha256=parent_hash osParentSignature=parent_signature osParentSigner=parent_signer incident=incident_id act=action suser=actor_effective_username

Example 4.

```
end=timestamp shost=endpoint_name deviceFacility=facility cat=category externalId=external_id request=request cs1=initiated_by_process cs1Label=Initiated by (constant string) cs2=initiator_commande cs2Label=Initiator CMD (constant string) cs3=signature cs3Label=Signature (constant string) cs4=cgo_name cs4Label=CGO name
```



```

name (constant string) cs5=cgo_command cs5Label=CGO CMD (constant string) cs6=cgo_signature cs6Label=CGO Signature (constant string) dst=destination_ip
dpt=destination_port src=source_ip spt=source_port fileHash=file_hash filePath=file_path targetprocesssignature=target_process_signature tenantname=tenant_name
tenantCDLid=tenant_id CSPaccountname=account_name initiatorSha256=initiator_hash initiatorPath=initiator_path osParentName=parent_name osParentCmd=parent_command
osParentSha256=parent_hash osParentSignature=parent_signature osParentSigner=parent_signer incident=incident_id act=action suser=actor_effective_username

```

#### 2.5.6.2.2 | Management Audit log notification format

##### Abstract

An email account or a syslog receiver are the notification channels through which the Management Audit log is communicated.

Cortex AgentiX forwards the Management Audit log to these external data sources:

- **Email account:** Sent according to the settings you configured. For more information, see Configure notification forwarding.
- **Syslog receiver:** Sent in a CEF format RFC 5425 according to the following mapping:

Section	Description
Syslog header	<9>: PRI (considered a priority field)1: version number2020-03-22T07:55:07.964311Z: timestamp of when issue /log was sentcortexxdr: host name
CEF header	HEADER/Vendor="Palo Alto Networks" (as a constant string)HEADER/Device Product="Cortex XDR" (as a constant string)HEADER/Device Version= Cortex XDR version (2.0/2.1....)HEADER/Severity=(integer/0 - Unknown, 6 - Low, 8 - Medium, 9 - High)HEADER/Device Event Class ID="Management Audit Logs" (as a constant string)HEADER/name = type
CEF body	suser=user end=timestamp externalId=external_id cs1Label=email (constant string) cs1=user_email cs2Label=subtype (constant string) cs2=subtype cs3Label=result (constant string) cs3=result cs4Label=reason (constant string) cs4=reason msg=event_description tenantname=tenant_name tenantCDLid=tenant_id CSPaccountname=csp_id

Example 5.

```

3/18/2012:05:17.567 PM<14>1 2020-03-18T12:05:17.567590Z cortexxdr -- CEF:0|Palo Alto Networks|Cortex XDR|Cortex XDR x.x |Management Audit
Logs|REPORTING|6|suser=test end=1584533117501 externalId=5820 cs1Label=email cs1=test@paloaltonetworks.com cs2Label=subtype cs2=Slack Report cs3Label=result
cs3=SUCCESS cs4Label=reason cs4=None msg=Slack report 'scheduled_1584533112442' ID 00 to ['CUXM741BK', 'C01022YU00L', 'CV51Y1E2X', 'CRK3VASN9'] tenantname=test
tenantCDLid=11111 CSPaccountname=00000

```

## 3 | Cortex AgentiX Configuration

##### Abstract

Configure Cortex AgentiX to match your use case.

As soon as you have completed onboarding with Cortex AgentiX, you can start configuring the tenant to match your use cases.

### 3.1 | Configure Cortex AgentiX

##### Abstract

Configure engines, playbooks, scripts, dashboards, etc., for your use case.

The following table describes how to configure the Cortex AgentiX tenant.



Section	Details	See More
Configure content	<p>Use the Data Onboarder wizard to install integration instances, playbooks, and dashboards.</p> <p>In addition, use the Marketplace to manage content packs and set up notifications.</p> <p>Configure integrations, including fetching issues, managing credentials, troubleshooting, and more.</p>	See topic
Agent Configuration	Create and edit agents and actions in the Agents Hub.	See topic
MCP Server	Set up the Cortex MCP Server to leverage Cortex's powerful features directly into your Large Language Model (LLM) apps.	See topic
Uses and roles	Configure and manage users, roles, and user groups. Assign roles and set up authentication for users.	See topic
Engines	If you have not done so already, you can configure and manage engines, such as using an engine as a web proxy and for load balancing.	See topic
Cases and Issues configuration	Learn how to customize cases and issues, domains, and create correlation rules	See topic
Automations	Learn how to customize and create playbooks and scripts, including creating tasks, sub-playbooks, and polling.	See topic
Jobs	Run playbooks based on certain events or on a specific time and date.	See topic
Lists	Create lists and add them to playbooks, scripts, and automation exclusion policies.	See topic
Cortex AgentiX XQL	Cortex AgentiX allows you to form complex queries against data stored in the tenant.	See topic
Data management	Learn how to create data model rules and manage datasets.	See topic
Dashboards and reports	View, edit, and create dashboards and reports, including widgets. View Command Center and predefined dashboards.	See topic

After you have configured Cortex AgentiX, analysts can start investigating issues and indicators.

## 3.2 | Content configuration

### Abstract

Learn how to manage content in Cortex AgentiX including adding a data source, downloading a content pack, and configuring an integration.

In Cortex AgentiX you can configure content by doing the following:

- Define a data source
- Install a content pack from Marketplace and then configure the integration.

### 3.2.1 | Configure content using Data Sources

#### Abstract

Use the Data Onboarder to install the integration instance and enable your integration. You can also set up content such as playbooks and dashboards in one place.



On the Data Sources page, when you add a data source, the Data Onboarder wizard guides you through installing the integration instance and enables you to configure your integration. Cortex AgentiX automatically downloads and installs the required content packs for the integration, and sets up and recommends additional beneficial content, such as playbooks and dashboards relevant to the specific data source.

You can also manage the configured instances by editing, deleting, enabling, or disabling them. On the Data Sources page, after locating the required data source

### 3.2.1.1 | Manage instances

You can manage the instances configured for a data source on the Data Sources & Integrations page. You can edit, delete, enable, or disable instances, and refresh log data.

1. Navigate to Settings â Data Sources & Integrations.
2. Find an instance by clicking on a data source name or using the Search field.
3. In the row for the instance name, take the required action:

Action	Instructions
Enable or disable an Instance	Select or deselect the Enable checkbox.
Edit an Instance	<ol style="list-style-type: none"><li>a. Click the Edit icon.</li><li>b. In Edit Data Source, you can update the values in the Connect and Collect sections. The options under Recommended Content are view only.</li><li>c. Test the configuration.</li><li>d. Connect the updated Instance.</li></ol>
Delete an Instance	<p>Click the Delete icon.</p> <p>If you delete all the instances for a Data Source, the Data Source is not listed on the Data Sources &amp; Integrations page.</p>

### 3.2.1.2 | Add a new data source or instance

#### Abstract

Use the Data Source Onboarder to add a new data source or instance in Cortex AgentiX.

You can add a new data source with the Data Source Onboarder. The Onboarder installs the data source, sets up an instance, configures playbooks and scripts, and other recommended content. The Onboarder offers default (customizable) options and displays all configured content in a summary screen at the end of the process.

1. Navigate to the Settings â Data Sources & Integrations page.
2. Select one of the following options:
  - Add a new data source: Click + Add New.
  - Add a new data source integration instance: Select an existing data source and click Add Instance. Then skip to Step 4.
3. Select a data source to onboard and click Add.

Hovering over a data source displays information about the data source and its integrations. Data sources that are already integrated are highlighted green and show Connect Another Instance. To see details of existing integrations, click on the number of integrations.

The data sources are drawn from the Marketplace, Custom Collectors, and integrations. If you search for a data source and No Data Sources Found, click Try searching the Marketplace, to view the marketplace page prefiltered for your search. If there are no available options in the Marketplace, you can use one of the Custom Collectors to build your own.

#### NOTE:



- If a data source contains multiple integrations, the integration configured as the default integration will be used by the Data Onboarder. The default integration of the content pack is indicated in each content pack's documentation. The other integrations are available for configuration in the Data Sources & Integrations page after installing the content pack.
- Not all content packs are supported.
- When adding XDR data sources, the Data Source Onboarder is not available. However, you can still enable the data source; Cortex AgentIX creates an instance and lists it on the Data Sources & Integrations page.

4. In the settings configuration pane, complete the mandatory fields in the Connect section.

For more information about the fields, click the question mark icon.

5. (Optional) Under Collect, select Fetched alerts and complete the fields.

6. Under Recommended Content, review and customize the options.

The items in this section are content-specific. Some options are view only, and others are customizable. Click on each option for more information:

- Classifiers & Mappers
- Data Normalization: Parsing rules and data models
- Correlations: Correlation rules included in the pack
- Automation: Playbooks and Scripts included in the pack.

You can select the Playbooks and Scripts that you want to enable. By default, recommended options are selected. Any unselected content is added as disabled content. Depending on the selected playbook, some scripts are mandatory.

- Dashboards & Reports: Recommended dashboards, widgets, and reports

#### **NOTE:**

- If you are adding a new instance to an existing data source, these options are View only.  
You can adjust the view-only options on the relevant page in the system, for example Correlations, Playbooks, or Scripts.
- Cortex AgentIX automatically installs content packs with required dependencies and updates any pre-installed optional content packs. You can also Select additional content packs with optional dependencies to be configured during connection.

7. Test the configuration.

If the test fails, you can Run Test & Download Debug Log to debug the error.

8. Connect the data source.

9. Review the configuration in the summary screen.

If errors occurred during the test, you can click See Details and Back to Edit to revise your configuration. For advanced configuration, click on any item to open a new window to the relevant page in the system (for example, Correlations or Playbooks) filtered by the configuration.

10. Click Finish to return to the Data Sources & Integrations page.

#### **3.2.1.3 | Configure integration permissions**

##### **Abstract**

Integration permissions enable you to restrict running commands to specific roles in integrations.

You can use role-based access control (RBAC) to restrict running commands to specific roles at the integration instance level. If you have multiple instances of the same integration, you can assign different roles (permission levels) for the same command in each instance.

For example, you may want limit the roles that can run potentially harmful commands, such as the ability to isolate endpoints.

Users who do not have permission to run a command cannot do the following:

- Run the command from the CLI.
- Complete pending tasks in a Work Plan that uses the restricted command.
- Edit arguments for playbook tasks that use the restricted command.
- Select the command when editing a playbook.
- Leverage the restricted command when executing a reputation command, such as IP, Domain, and File.

If you have multiple instances of the same integration, you can assign different roles (permission levels) for the same command in each instance.

To view or edit integration permissions:



1. Go to Settings → Configurations → Data Collection → Integration Permissions.

You can see a list of all enabled integrations.

2. Select the integration.

You can see the following:

- INSTANCE: Lists all instances for the integration.
- COMMANDS: Lists all commands for the integration.
- PERMITTED ROLES: Lists the roles that have permission to run the command. Default is No Restrictions.

3. For a specific command, restrict the roles that can run the command.

1. Go to the relevant command.

2. Click Edit.

3. In the PERMITTED ROLES, column, select the roles that you want to allow running the command.

4. Save the integration permissions.

### 3.2.2 | Marketplace

Abstract

Use the Marketplace, a centralized content portal, to download and manage content packs in Cortex AgentiX.

Marketplace is a centralized content portal enabling you to manage content in Cortex AgentiX. Content is organized into content packs created by different contributors such as Palo Alto Networks, Partners, and MSSPs. Download a content pack to suit your use case.

You can view Marketplace content packs from within Cortex AgentiX or at <https://cortex.marketplace.pan.dev/marketplace/>.

#### 3.2.2.1 | Cortex Marketplace

Abstract

Search the Cortex Marketplace and find content. Search by use cases, integrations, and categories.

Content in Marketplace is organized into content packs to support specific security orchestration use cases. Content packs are created by Palo Alto Networks, technology partners, contributors, and customers.

In Marketplace, content includes the following:

Content	Description
Actions	Actions wrap diverse capabilities (such as playbooks, scripts, and commands) to make them accessible and executable by an agent.
Classifiers	Classification determines the type of issue/indicator that is created for events ingested from a specific integration. You create a classifier and define that classifier in an integration. Mappers map the fields from your third-party integration to the fields in your issue/indicator layouts.
Correlation Rules	Analyzes the correlation of multiple events from multiple sources by using the Cortex AgentiX XQL-based engine for creating these correlation (scheduled) rules. Issues can then be triggered based on these rules with a defined time frame and schedule.
Dashboards	Dashboards consist of visualized data powered by fully customizable widgets, which enable you to analyze data from inside or outside Cortex AgentiX, in different formats such as line charts, tables, text, etc.



Content	Description
Data Model Rules	<p>Data Model rules enable you to normalize logs for out-of-the-box analytics and data enrichment. This allows you to do the following:</p> <ul style="list-style-type: none"> <li>Map 3rd-party data to a consolidated schema with predefined data types.</li> <li>Enjoy auto-complete and mapping suggestions.</li> <li>Map multiple datasets to one Data Model.</li> </ul> <p>Some content packs contain out-of-the-box default Data Model Rules.</p>
Indicator types and fields	<p>Indicators are categorized by indicator type, which determines the indicator layout and fields that are displayed and which scripts are run on indicators of that type.</p>
Integrations	<p>You can define the following integrations:</p> <ul style="list-style-type: none"> <li>(SOAR) Automation: Add your 3rd-party security and alert management vendors, which can then trigger events from these integrations that become issues in Cortex AgentiX. Once the issues are created, you can run playbooks on these issues to enrich them with information from other products in your system, which helps you complete the picture.</li> <li>Collection (SIEM): Add integrations that collect raw events, such as logs. These integrations are separate from automation integrations so that you can add a collection integration that requires read permissions without having to add automation (read and write permissions).</li> </ul>
Issue types and fields	<p>All issues that are ingested into Cortex AgentiX are assigned an issue type when they are classified. After you classify the issue, you can then map the relevant fields to the issue.</p> <p>Issue types contain fields that are relevant to the issue type.</p>
Layouts and layout rules	<p>Enables you to add rules, which define the layout of issues and notifications.</p> <p>When installed, the layout rules are enabled and added as Default Rules. When deleted, all related layout rules (including all Rule sections) are removed from the Default Rules tab.</p>
Parsing rules	<p>Enables you to add rules, which remove non-required data for analytics, hunting, or regulation, reduce data storage costs, pre-process all incoming data, etc.</p> <p>When installed, the parsing rules are enabled and added as Default Rules. When deleted, all related parsing rules (including all Rule sections) are removed from the Default Rules tab.</p>
Playbooks	<p>You can automate many security processes, including handling investigations and managing tickets and security responses that were previously handled manually. When an issue is ingested, the playbook runs and an issue is created.</p>
Reports	<p>Reports contain statistical data in the form of widgets (from a dashboard), which enable you to analyze data from inside or outside Cortex AgentiX, in different formats such as line charts, tables, text from information, etc.</p>
Scripts	<p>Perform specific actions and are comprised of commands, which are used in playbook tasks and when running commands in the issue War Room.</p>

Cortex AgentiX supports free content packs, which are either Cortex AgentiX or partner-supported content packs. You can restrict a user role from managing content packs in Marketplace when defining/editing user roles.

In Marketplace, you can browse all content packs (including installed content) or view only installed content packs.

You can search for content packs by entering text in the search bar and selecting the relevant content pack from the search results.

You can sort content packs by latest update, best match, recommended, number of downloads, and filter according to the following criteria:



- **Integrations:** Filter according to the integration included in the content pack.
- **Categories:** Filter according to content pack categories, such as Messaging, and Forensics & Malware Analysis
- **Published:** Filter according to whether published by Cortex AgentiX or by Cortex AgentiX technology partners.
- **Content Pack Includes:** Filter according to the content of the content pack, such as scripts, integrations, playbooks, and actions.
- **Tags:** Filter according to tags, such as Issues, Actions, Network, and Security.
- **Types:** Filter according to Collection or TIM.

When clicking a content pack you can view detailed information including content that it installs (such as scripts, playbooks, and integrations), dependencies (what content packs are required or optional) and version history (including whether you want to roll back to earlier versions).

You can view Marketplace content packs from within Cortex AgentiX (go to Settings → Configurations → Marketplace) or at Cortex Developer Docs Marketplace.

### 3.2.2.2 | Content packs

#### Abstract

Download content packs in Marketplace for your use case.

Content packs are created by Palo Alto Networks, technology partners, consulting companies, MSSPs, customers, and individual contributors. Content packs may include a variety of different components, such as integrations, scripts, playbooks, and widgets, grouped together to address a specific use case. Content packs are free and can be used by all customers.

You can view Marketplace content packs from within Cortex AgentiX (go to Settings → Configurations → Marketplace) or at Cortex Developer Docs Marketplace.

#### Pre-installed content packs

Cortex AgentiX comes with a number of pre-installed content packs that cover many common use cases. Pre-installed content packs include, but are not limited to:

- Common Scripts, Common Widgets, Common Playbooks, Common Types, Common Reports, Common Dashboards

These content packs provide important tools and building blocks you can use to customize your playbooks and workflows in Cortex AgentiX. The Common Scripts content pack, for example, includes scripts that convert file formats, fetch indicators from a file, export context data, send emails, and more.

- VirusTotal

Provides integration with the popular Virus Total service to analyze suspicious files, domains, IPs and URLs to detect malware and other security breaches.

#### Recommended content packs

In addition, we recommend reviewing if you require the following popular content packs:



## Suggested Use Cases to start with



- Phishing

Create and respond to phishing issues based on user reports.

- Cortex XDR by Palo Alto Networks

Automate Cortex XDR incident response. Includes custom Cortex XDR incident views and layouts to aid analyst investigations.

- Atlassian Jira

Manage Jira tickets directly from Cortex AgentiX, enrich them with Cortex AgentiX data, and mirror information between Jira tickets and Cortex issues.

- ServiceNow

Manage ServiceNow tickets directly from the Cortex AgentiX and enrich them with Cortex AgentiX data, and mirror information between ServiceNow tickets and Cortex issues.

- PAN-OS by Palo Alto Networks

Manage Palo Alto Networks Firewall and Panorama, from Cortex AgentiX.

- A collaboration integration, such as Microsoft Teams or Slack to send messages and notifications to your team.

**NOTE:**

Cortex AgentiX includes a built-in default mail sender. You also have the option of installing a different mail sender content pack, such as Microsoft Exchange Online.

### 3.2.2.3 | Content Pack Support Types

#### Abstract

Types of content packs support - Cortex supported, Partner-Supported, Developer-Supported, Community-Supported.

Marketplace includes the following content pack support types:

#### Cortex AgentiX-Supported content packs

Applies only to content packs published by Palo Alto Networks. These content packs are supported and maintained by Palo Alto Networks according to the Palo Alto Networks End User Support Agreement.

**NOTE:**

Palo Alto Networks is not liable for and does not warrant or support any content pack produced by a third-party publisher.

Palo Alto Networks does not support content packs that do not have official available documentation.

#### Partner-Supported content packs



Applies to content packs published by Cortex AgentiX Technology Partners. Support and maintenance is provided by the Technology Partner, whose contact information appears in the content pack details.

Cortex AgentiX Technology Partners are required to join the industry-standard support framework, TSANet, to deliver support to our mutual customers. Customers engage directly with the partner for support and maintenance of the partner-supported content pack.

#### Developer-Supported content packs

Applies to content packs published by third-party developers. Support and maintenance is provided by the publishing developer, whose contact information appears in the content pack details.

Customers engage directly with the publishing developer. Support and maintenance is provided voluntarily by the publishing developer. Additional information from the user community may be available at Cortex XSOAR Live Discussions.

#### Community-Supported content packs

Applies to content packs published by Palo Alto Networks or third-party developers. No support or maintenance is provided by the publisher for these content packs.

Palo Alto Networks ensures that these content packs are updated to use the latest and most secure Docker images through an automated process. However, functionality may not be fully tested. We recommend fully testing and reviewing Community content packs before updating production systems.

### 3.2.2.4 | Cortex AgentiX content

#### Abstract

The type of content in Cortex AgentiX

In Cortex AgentiX, content includes individual content entities that you create such as individual playbooks and issue fields, preinstalled content packs, and content packs that you download from Marketplace.

### 3.2.2.5 | Manage content packs

#### Abstract

Install, delete, update, and revert content packs.

You can install, delete, update, and revert content packs. Before you install a content pack, you should review the content pack to see what it includes and the various dependencies. The following is the information you can view:

- **Details:** General information about the content pack such as installation, content, version, author, and status.
- **Content:** The content to be installed, such as scripts or integrations.
- **Dependencies:** Details of any required content packs and optional content packs that may need to be installed with your content pack.
- **Version History:** View the currently installed version, earlier versions, available updates, and revert if required.

#### Dependencies

In Cortex AgentiX content packs, some objects are dependent on other objects. For example, an issue may be dependent on a playbook, an issue type, and an issue field. A script may be dependent on another script, or an integration. When you place a content pack in your cart, mandatory dependencies including required content packs are added automatically to ensure that the content pack installs correctly.

Optional content packs are used by the content pack you want to install, but are not necessary for installation. When you place a content pack in your cart, you can choose which optional content pack to install. When you install optional content packs, mandatory dependencies in the optional content pack are automatically included.

#### **NOTE:**

Optional content packs that are already installed are treated like they are required content packs to preserve content integrity.

#### Install a content pack

You can only install one content pack at a time. Cortex AgentiX automatically adds any content that is required to install the content pack. You can also add any optional content packs that use the content pack you want to install.

If you receive an error message when you try to install a content pack, you need to fix the error before installing. If a warning message is issued, you can still download the content pack, but you should fix the problem; otherwise, the content may not work correctly.

#### **NOTE:**

Cortex AgentiX includes a built-in default mail sender. You also have the option of installing a different mail sender content pack, such as Microsoft Exchange Online.



1. Go to Settings â€“ Configurations â€“ Marketplace â€“ Browse and locate the content pack you want to install.

2. Click the required content pack and review the contents.

3. Click Install to add the content pack to the Cart.

4. (Optional) If the content pack includes optional content, select the content packs you want to add.

The Cart displays the number of items you are installing, including any required content packs. You can log in and out, but the content packs remain in the Cart until you click either Empty cart or Install.

5. Click Install.

6. After installation, click Refresh content.

**NOTE:**

In addition to content packs that you install from Marketplace, related content packs are automatically downloaded when you adopt playbooks or edit tasks that require content items such as scripts or integrations.

Update a content pack

Content packs are updated for bug fixes, enhancements, and more. Marketplace is updated every 2 hours and when there is an update available for a content pack, you will see a notification in the Installed Content Packs tab in Marketplace.

In the Version History tab of a content pack, you can see the currently installed version, earlier versions, and available updates. You can revert to a previous version of a content pack if required.

All dependent content packs update automatically with the content pack.

**TIP:**

You can also find content packs that require updates by going to Settings â€“ Data Sources & Integrations and filtering by Pack Version = Update Available. If you click on an integration in the filtered list, there is a link to the content pack in Marketplace for updates.

**NOTE:**

Third-party product integrations are developed and tested against a specific product version. For products that are on-prem or cloud-based with specific API versions, the version developed and tested against will be included in the integration's documentation. Newer versions of the product are not always immediately tested, and it is expected that products maintain API compatibility upon release of newer product versions. When upgrading to a newer product version, it is highly recommended to test the integration in a dev environment before deploying to production.

**CAUTION:**

If you want to downgrade, any content that depends on the content pack including any customizations may be deleted if it does not exist in the target content pack version.

1. In the Show field of the Installed Content Packs tab, select Update available to display the content packs that are available to update.

2. Click the content pack you want to update.

3. In the Version History tab of the content pack, view the available updates.

4. Click Update. If there is more than one update available, click the version to update.

If you choose to install the latest version it includes the previous version. If you have made any customizations these are included in any update. If any dependencies require updating, these are automatically added.

5. Click Install.

6. After the content pack installs, click Refresh content.

Revert a content pack

You can revert to an earlier version of an installed content pack. Items that are not included in the version are also deleted, such as detached playbooks or scripts that use other scripts from the content pack. This may cause other content packs to stop working.

1. In the Installed Content Packs tab, click the content pack you want to revert.

2. In the Version History tab, select the version to which you want to revert.

3. Click Revert to this version. The version will be added to your Cart.

4. In the Cart, click Downgrade.

Delete a content pack

When you delete a content pack, all content is deleted, including all detached and customized content.

**CAUTION:**

If another content pack is dependent on the content pack you want to delete, it may break the other content pack. You can reinstall the content pack, but you cannot restore detached and customized content.



1. Go to Settings → Configurations → Marketplace → Installed Content Packs.
2. In the Content Packs Library section, search for the content pack and select the content pack you want to delete.
3. Click the trash can icon.
4. Review the warning message and click Delete.

### 3.2.2.6 | Marketplace FAQs

#### Abstract

Frequently Asked Questions about Cortex AgentiX Marketplace Content

##### Should Marketplace content always be updated?

Marketplace updates are a source for bug fixes and provide new commands for integrations and scripts. It's best practice to update content packs to the newest available version. If you encounter any issue with content updates, you can revert to a previous version with one click.

##### When can Marketplace content be updated?

You can update content while the system is in use. If a playbook, for example, is running on an issue while you update that playbook, the original version of the playbook will continue to run without a problem. If the playbook includes an integration command that has been updated, and the update occurs before the playbook reaches this task, the new version of the integration command will be used.

##### When should content items be duplicated versus detached?

To edit a content item, the item must be detached or custom content. When content items are detached, they do not receive updates from Marketplace. There are two options for editing content items:

- Detach content items (such as playbooks and automations) and edit the content items. If you want to receive content updates in the future, you can reattach the content item, but the modifications you made while the item was detached will be overwritten with the content update.
- Duplicate the content item and edit the copy. When a content item is duplicated it becomes a custom content item, and therefore will not receive updates, but you can view updates to the original content item.

##### How does Marketplace content differ from custom content?

After Marketplace content is installed you can detach or duplicate the content and customize the content as needed. Custom content is, by definition, detached and does not receive updates.

##### How can content updates be rolled back? Are dependencies automatically rolled back as well?

You can view all versions of a content pack in Marketplace and revert to earlier versions there. When you revert a content pack, only the content pack is reverted, not the pack dependencies.

### 3.2.2.7 | Content changes when upgrading Cortex AgentiX versions

#### Abstract

Content updates when upgrading Cortex AgentiX versions.

Cortex AgentiX will be upgraded automatically approximately every 3 months. During the upgrade, the following core content packs may automatically upgrade to a newer version:



- Aggregated Scripts
- Atlassian Jira
- AWS
- Azure
- Base
- Common Playbooks
- Common Scripts
- Common Types
- Core
- Core Issue Fields
- Cortex Lock
- Cortex Response And Remediation
- Cortex REST API
- Filters And Transformers
- Generic Export Indicators Service
- GCP
- HelloWorld
- Image OCR
- Microsoft Teams
- MITRE ATT&CK Feed V2
- Prisma Cloud
- Rasterize
- ServiceNow
- Slack
- Threat Intelligence Management
- Unit 42 Threat Intelligence by Palo Alto Networks
- VirusTotal
- Whois
- WildFire by Palo Alto Networks

### 3.2.2.8 | Content pack contributions

#### Abstract

You can create content packs for submission to the Cortex Marketplace.

Contributions are content packs that you create for Marketplace, which are submitted to the Cortex AgentiX content development team for review and approval. After approval, these content packs are uploaded to Marketplace, and are shared and installed like any other content pack. When creating new content you want to share, such as playbooks, scripts, issue types, and integrations, or when updating content, you can create a GitHub pull request to the public content repository, to submit your content for review.

#### Review process

The review process consists of the Cortex AgentiX team checking that your contribution meets code, documentation, naming, and other standards. You receive a form to complete asking for more information, such as certification, contact details, etc. The Cortex AgentiX team will be in touch with you during the review process.

During the review process, you may be asked to make changes in the code, or for more data, metadata, dependencies, documentation, support, and certification model, etc. You can anonymize your name if required.



When your contribution is approved, it is uploaded to Marketplace where other Cortex AgentiX users can view, download, and rate it. We encourage you to learn more about the contribution process.

### 3.2.3 | Integrations

#### Abstract

Set up an integration instance and start ingesting cases/indicators.

Integrations are mechanisms through which Cortex AgentiX connects and communicates with other products. These integrations can be executed through REST APIs, webhooks, and other techniques. Integrations enable you to orchestrate and automate SOC operations.

#### Integrations installed from a content pack

Integrations are included in content packs, which you download and install from Marketplace (go to Settings → Configurations → Marketplace). After you download and install a content pack that includes an integration, you need to configure the integration by adding an instance. You can have multiple instances of an integration, for example, to connect to different environments. Additionally, if you are an MSSP and have multiple tenants, you could configure a separate instance for each tenant.

#### NOTE:

- Some integrations can be downloaded directly without having to initially download a content pack from Marketplace. For more information, see Configure content using Data Sources.
- In addition to content packs that you install from Marketplace, related content packs are automatically downloaded when you adopt playbooks or edit tasks that require content items such as scripts or integrations.

Cortex AgentiX comes out-of-the-box with integrations to help you onboard, such as:

- Mail Sender

Sends email notifications to users.

- Generic Export Indicators Service

Provides an endpoint with a list of indicators as a service for the system indicators. For more information about how to set up the integration, see Export indicators using the Generic Export Indicators Integration.

- Palo Alto Networks WildFire Reports

Generates a Palo Alto Networks WildFire PDF report. For more information, see Palo Alto Networks WildFire Reports.

- Rasterize

Converts URLs, PDF files, and emails to an image file or PDF file. For more information, see Rasterize.

#### Create an integration

You can create an integration, by adding parameters, commands, arguments, and outputs as well as writing the necessary integration code. You should have a working Cortex AgentiX tenant and programming experience with Python.

1. Navigate to the Settings → Data Sources & Integrations page and click + Add New.
2. In the Add Data Source or Integrations page click Create Integration and select either Import File or Create from Template.
3. If you select Import File drag and drop the download the relevant integration file. If you select Create from Template, provide the integration code and settings.

For more information about how to create an integration, including an example, see Create an Integration.

#### Configure an integration

On the Data Sources & Integrations page, after either downloading or creating an integration you can do the following:

Option	Description
Add instance	Configure an integration instance to connect and communicate with other products. For more information, see Add an integration instance.  After configuring the instance, you can also enable/disable the integration instance, copy the instance, and view the integration fetch history.



Option	Description
View integration's source	View the integration settings and source code.
Edit integration's source	Edit the integration settings and source code. For more information about editing the integration's source code, see Create an Integration.  <b>NOTE:</b> If the integration was installed from a content pack, you need to duplicate the integration before editing.
Duplicate integration	If you want to change the source code, and settings, or download the integration, you need to duplicate the integration.
Delete	Although you can't delete an integration installed from a content pack (unless a duplicate), you can delete an integration instance.
Always / On Demand	For each integration instance, you have the option of setting the instance to be used only On Demand, when it is specified with the <b>using</b> argument in a playbook or the CLI. By default, the settings is Always and the integration instance is used whenever the integration is called.
Download the integration	Download the integration in YAML format. You can also upload an integration.  <b>NOTE:</b> If the integration was installed from a content pack, you need to duplicate the integration before downloading.
Version History	If the integration is a duplicate or you create your own integration, you can see the changes in the integration.

You can view all the integration changes (the last 100 changes) by clicking the Version History button.

#### Using integration commands

The command line interface (CLI) enables you to run system commands, integration commands, scripts, etc from the Cases War Room, Issues War Room, or Playground CLI. The CLI auto-complete feature allows you to find relevant commands, scripts, and arguments.

Cortex AgentIX uses the "!" such as `!ad-create-user username=[name of user]`

Under each integration, you can view a list of commands.

#### NOTE:

Integration commands are only available when the integration instance is enabled. Some commands depend on a successful connection between Cortex AgentIX and third-party integrations.

You can run the CLI commands in the Playground or in a case/issue War Room. The Playground is a non-production environment where you can safely develop and test automation scripts, APIs, commands, etc. It is an investigation area that is not connected to a live (active) investigation.

When running the command, the results are returned in the War Room or Playground and also in a JSON format in Context Data.

#### TIP:

In the Playground, you can clear the context data, if needed, which deletes everything in the Playground context data, but does not affect the actual issue or case. To clear the context, run `!DeleteContext all=yes` from the CLI or click Clear Context Data while viewing the context data.

#### 3.2.3.1 | Manage API keys

API keys are used to manage and secure API interactions. An API key is essentially a unique string of alphanumeric characters that acts as a credential, allowing a specific user or application to access and interact with a particular API. When you request data or perform an action through an API call, you must include this API key in the header. Cortex AgentIX then verifies the key's authenticity and, if valid, grants the requested access.

##### How to create an API key

1. Select Settings → Configurations → Integrations → API Keys → New Key.
2. In the Role tab, perform for the following:



- a. Under Security Level, select the type of API Key you want to generate: Advanced or Standard. The Advanced API key hashes the key using a nonce, a random string, and a timestamp to prevent replay attacks. cURL does not support this but it is suitable with scripts.
  - b. Under Role, select the desired level of access for this key. You can select from predefined roles or custom roles. Roles are available according to what was defined in either the Cortex Gateway or Cortex AgentX Access Management. You can view the configuration of the role selected by expanding the sections under Components. For more information, see Assign user roles and groups.
  - c. (Optional) Under Comment, provide a comment that describes the purpose of the API key.
  - d. (Optional) If you want to define a time limit on the API key authentication, select Enable Expiration Date, and select the expiration date and time. You can track the expiration date of each API key in the API Keys page. In addition, Cortex AgentX displays a API Key Expiration notification in the Notification Center one week and one day prior to the defined expiration date.
3. (Optional) To configure and manage granular scoping for Scope-Based Access Control (SBAC), click the Scope tab, and under Scope Definition, expand the scoping areas that you want to grant the user role access to for this API by clicking the chevron icon (>) beside the scoping area title. The following table explains the options available to configure:

**IMPORTANT:**

Before configuring, ensure that you review Understand scoping in the Manage user scope section.

Scoping Area	Granular Scoping Configurations
Assets	<p>Set the Scope by selecting one of the following:</p> <ul style="list-style-type: none"> <li>• No assets: No asset is accessible.</li> <li>• All assets: Defines access to all assets.</li> <li>• Select asset groups: Defines access to the specific assets associated with the Asset Groups selected, and to view all their related cases, issues, and findings for these specific assets and Asset Groups. Under Select asset groups, define the specific asset groups that you want to grant access. Only Asset Groups relevant for scoping are listed, which are asset groups that are using only the asset attributes listed in Manage user scope (under Understand scoping â Scoping Areas â Assets).</li> </ul> <p>The scoping of assets also affects the scoping of cases, issues, and findings.</p> <p><b>NOTE:</b></p> <p>Visibility of Security domain Issues that refer to assets with agents is controlled by the Endpoints scoping configuration.</p>
Cases and Issues	<p>Set the Scope by selecting one of the following:</p> <ul style="list-style-type: none"> <li>• No cases and issues: Defines access to no cases and issues.</li> <li>• All cases and issues: Defines access to all cases and issues. Users can view cases or issues referencing assets within their scope. Use the Assets section to define which assets are in scope.</li> <li>• Select domains: Defines access to the domains selected to view their related cases and issues. Under Select domains, define the specific domains that you want to grant access.</li> </ul> <p>Users can only view cases or issues referencing assets and endpoints within their scope. Use the Assets section to define which assets are in scope.</p> <p>When selecting All cases and issues or Select domains, you can separately configure access to issues and cases that lack an asset reference or where the referenced asset is not in All Assets and All Endpoints inventories. To provide access, select the Allow access to cases and issues that are not referencing known assets or endpoints checkbox.</p>
Endpoints	<p>Set the Scope by selecting one of the following:</p> <ul style="list-style-type: none"> <li>• No endpoints: Defines access to no endpoints with no ability to view their related agent management and enterprise policies.</li> <li>• All endpoints: Defines access to all endpoints with the ability to view their related agent management and enterprise policies. This configuration can impact the visibility of related Security domain Cases and Issues, but will not affect asset visibility.</li> <li>• Select specific (at least one required): Defines specific access to all endpoint groups by selecting Endpoint Groups or all endpoint tags by selecting Endpoint Tags to view their related agent management and enterprise policies. This configuration can impact the visibility of related Security domain Cases and Issues, but will not affect asset visibility.</li> </ul>



**IMPORTANT:**

By default, Enable Scope Based Access Control is disabled in Settings → Configurations → General → Server Settings, and granular scoping is not enforced. Before enabling SBAC, we recommend that an administrator or a user with Access Management permissions first ensures that the users, user groups, and API Keys defined in Cortex AgentiX are granted the required access by assigning the relevant scopes. For more information, see [Manage user scope](#).

4. Click Generate to generate the API key.
5. Copy the generated API key and click Done.

**IMPORTANT:**

You will not be able to view the API key again after you complete this step. Ensure that you copy the API key before closing the notification.

**Actions available on API Keys**

Below are some of the main pivot (right-click) options for actions available on each API key listed in the API Keys table. Only tasks that need further explanation are explained below.

Action	Description
View Examples	Copies the Python 3 example, so you can edit it to set up your own API calls.
Copy text to clipboard / Copy entire row	Copies the value of an API setting, such as the ID, to the clipboard by right-clicking the setting and selecting Copy text to clipboard. You can copy all the settings of an API key by right-clicking and selecting Copy entire row.
Filter API keys	Filters the API keys by selecting one of the filter options, such as Show rows 30 days prior to.... You can then adjust the filter options to filter the API keys according to all the available fields.

### 3.2.3.2 | Integration use cases

#### Abstract

Common integration use cases for Cortex AgentiX, including analytics and SIEM, authentication, case management, data enrichment, threat intelligence, forensics and malware.

The following categories are common use cases for Cortex AgentiX integrations. While this list is not meant to be exhaustive, it's a starting point to understand what use cases are supported by Cortex AgentiX and third-party integrations.

**Analytics and SIEM**

Top use cases:

- Fetch issues with relevant filters.
- Create, close, and delete issues/events/cases.
- Update issues - update status, assignees, severity, SLA, and more.
- Get events related to an issue/case for enrichment/investigation purposes.
- Query SIEM (consider aggregating logs).

These integrations usually include the Fetch Issues or Fetch Alerts option for an integration instance configuration. The integration may also include integration commands enabling you to list or retrieve issues or related information.

Analytics & SIEM integration Example: ArcSight ESM

**Authentication and Identity Management**

Top use cases:



- Use credentials from the authentication vault to configure instances in Cortex AgentiX. (Save credentials in: Settings → Configurations → Integrations → Credentials.) Integrations that use credentials from the vault should have the Switch to credentials option.
- Lock/Delete Account → Use an integration to lock/unlock a third-party account.
- Reset Account - Perform a reset password command for a third-party account.
- Lock an external credentials vault - in case of an emergency (if the vault has been compromised), allow the option to lock/unlock the entire vault via an integration.
- Step-Up authentication - Enforce Multi-Factor Authentication for an account.
- Create, update, and delete users.
- Manage user groups.
- Block users, force a change of passwords.
- Manage access to resources and applications.
- Create, update, and delete roles.

Authentication integration example: CyberArk AIM v2 (Partner Contribution)

#### Case Management

Top use cases:

- Create, get, edit, close a ticket or issue, and add and view comments.
- Assign a ticket/issue to a specified user.
- List all tickets, and filter by name, date, and assignee.
- Get details about a managed object, update, create, or delete.
- Add and manage users.

Case Management/Ticketing integration example: ServiceNow V2

#### Data Management and Threat Intelligence

Top use cases:

- Enrich information about different IOC types: Upload object for scan and get the scan results. (If there's an option to upload private/public, the default should be set to private.) Search for former scan results about an object to get information about a sample without uploading it yourself. Enrich information and scoring for the object.
- Add indicators to the system and search for existing indicators.
- Add indicators to the exclusion list.
- Calculate DBot Score for indicators.
- Enrich asset → get vulnerability information for an asset (or a group of assets) in the organization.
- Generate/trigger a scan on specified assets.
- Get a scan report including vulnerability information for a specified scan and export it.
- Get details for a specified vulnerability.
- Scan assets for a specific vulnerability.

Data Enrichment & Threat Intelligence integration example: Unit 42 Intelligence.

#### Email

Top use cases:



- Get message – download the email itself, retrieve metadata, and body.
- Download attachments for a given message.
- Manage senders – block/allow specified mail senders.
- Manage URLs – block/allow the sending of specified URLs.
- Encode/decode URLs in messages
- Release a held message when a gateway has placed a suspicious message on hold.

Email Gateway integration example: MimeCast v2

#### Endpoint

Top use cases:

- Fetch issues and events
- Get event details (from a specified alert)
- Quarantine a file
- Isolate and contain endpoints
- Update indicators (for example, network and hashes) by policy (can be block, monitor) – deny list
- Add indicators to the exclusion list
- Search for indicators in the system (see indicators and related issues/events)
- Download a file based on the hash and the path
- Trigger scans on specified hosts
- Update .DAT files for signatures and compare existing .DAT files to the newest one on the Cortex AgentX tenant
- Get information for a specified host (OS, users, addresses, hostname)
- Get policy information and assign policies to endpoints

Endpoint integration example: Palo Alto Networks Cortex XDR - Investigation and Response

#### Forensics and Malware Analysis

Top use cases:

- Submit a file and get a report (detonation)
- Submit a URL and get a report (detonation)
- Search for past analysis (input being a hash/URL)
- Retrieve a PCAP file
- Retrieve screenshots taken during analysis

Forensic and Malware Analysis example: Cuckoo Sandbox

#### Network Security

Top use cases:



- Create block/accept policies (source, destination, port), for IP addresses and domains
- Add addresses and ports (services) to predefined groups, create groups, and more
- Support custom URL categories
- Fetch network logs for a specific address for a configurable time frame
- URL filtering categorization change request
- Built-in blocked rule command for fast blocking
- If there is a Management Firewall, allow the option to manage policy rules through it
- Get/fetch issues
- Get PCAP file, packet
- Get network logs filtered by time range, IP addresses, ports, and more
- Create/manage/delete policies and rules
- Update signatures from an online source/upload + get the last signature update information
- Install policy (if existing)

Network Security Firewall integration examples: Tufin (Partner Contribution), Protectwise

#### Vulnerability Management

Top use cases:

- Enrich asset – get vulnerability information for an asset (or a group of assets) in the organization.
- Generate/trigger a scan on specified assets
- Get a scan report including vulnerability information for a specified scan and export it
- Get details for a specified vulnerability
- Scan assets for a specific vulnerability

Vulnerability Management integration example: Tenable.sc

#### 3.2.3.3 | Add an integration instance

##### Abstract

Set up an integration instance and start ingesting issues/indicators.

Configure an integration instance to connect and communicate with other products.

When you define an integration instance for your third-party security and incident management vendors, events triggered by this integration instance can become cases in Cortex AgentIX. When cases are created, you can run playbooks on them to enrich them with information from other products in your system. For indicators, you can enrich those indicators depending on the integration instance and add them to a case if required.

Although you can view the integration documents when adding an instance, the Developer Hub has more detailed information about the integrations, including commands, outputs, and recommended permissions. You can also see more information about content packs, playbooks, scripts, and Marketplace documentation.

##### **NOTE:**

This procedure describes how to add an integration instance from the Data Sources & Integration page. Some integration instances can also be configured on the Data Sources page. For more information, see [Add a new data source or instance](#).

Before you begin

- From Marketplace, download and install the relevant content pack, which includes your integration. Content packs containing integrations are also downloaded, in some cases, when you adopt playbooks and configure playbook tasks.
  - Consider whether you want to add credentials, which enable you to save login information without exposing usernames, passwords, certificates, and SSH keys. For more information, see [Manage credentials](#).
1. Navigate to Settings → Data Sources & Integrations and search for the integration for which you want to add an instance.
  2. Select the integration and click Add Instance.
  3. Add the parameters as required.



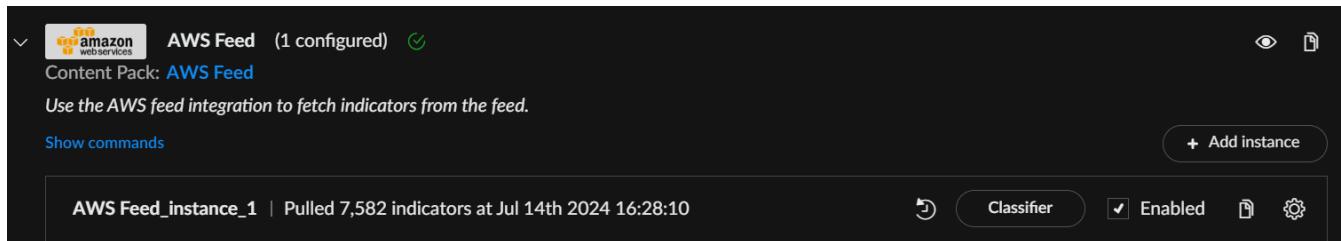
4. If you want to fetch issues, select the Fetches alerts.

For more information, see Fetch issues from an integration instance.

5. (Optional) To check that the integration instance is working correctly, click Test.

6. Save & Exit.

Expand the integration to see more details such as the number of pulled issues/indicators or error messages.



The screenshot shows the AWS Feed integration instance configuration page. At the top, it displays "AWS Feed (1 configured)" and "Content Pack: AWS Feed". Below this, a note says "Use the AWS feed integration to fetch indicators from the feed." A "Show commands" button is present. In the center, the instance name "AWS Feed\_instance\_1" and the message "Pulled 7,582 indicators at Jul 14th 2024 16:28:10" are shown. On the right, there are buttons for "Classifier" (selected), "Logs" (disabled), and "Enabled" (checked). Below the logs button are icons for copy, refresh, and settings.

You can also enable/disable the integration instance, copy the instance, and view the integration fetch history.

If you encounter an error, see Troubleshoot Integrations.

7. By default, the integration instance is used whenever the integration is called. If you want to only use the integration instance when specified with the `using` argument in a playbook or the CLI, change the integration instance setting from Always to On Demand. For example, you might have two instances of an integration and want to use one instance as the default and the other instance only for manual testing on demand.

8. (Optional) To manage access to specific commands, see Configure integration permissions.

Example 6.

In this example, you will set up the OnboardingIntegration.

If you have not done so, download the OnboardingIntegration content pack from Marketplace. Most integrations follow a similar configuration.

1. Go to Settings → Data Sources & Integrations and search for OnboardingIntegration.

2. Click Add Instance.

3. Add the number of issues to fetch per minute. By default, there is a maximum of 5 issues per minute.

4. Add the maximum number of issues to create. By default, there is a maximum number 10 issues to create.

5. Add the number of issues you want to create in minutes.

6. Set the Alerts Fetch Interval. By default, the issues are fetched every minute.

7. Select whether to run on an engine.

8. When troubleshooting the instances, adjust the default setting from off to a higher debugging level.

9. Select Fetches alerts to start ingesting issues.

For all integrations, we recommend only fetching issues when everything is set up. When enabled, Cortex AgentiX searches for events that occurred within the time frame set for the integration, which is based on the specific integration. The default is 5 issues per minute.

#### NOTE:

In some integrations, a classifier, an issue type, and mapper fields are included.

10. Test and Save & Exit.

#### 3.2.3.4 | Fetch issues from an integration instance

Abstract

Configure a third-party integration instance to fetch issues into Cortex AgentiX cases for investigation.

You can poll third-party integration instances for events and turn them into Cortex AgentiX issues (fetching). Many integrations support fetching, but not all support this feature. You can view each integration in the Developer Hub.

When setting up an instance, you can configure the integration instance to fetch events. You can also set the interval for which to fetch new issues by configuring the Issue Fetch Interval field. The fetch interval default is 1 minute. This enables you to control the interval in which an integration instance reaches out to third-party platforms to fetch issues into Cortex AgentiX.



**NOTE:**

- In some integrations, the Issue Fetch interval is called Feed Fetch Interval.
- If the integration instance does not have the Issue Fetch Interval field, you need to add this field by editing the integration settings. If the integration is from a content pack, you need to create a copy of the integration. Any future updates to this integration will not be applied to the copy integration.
- If you turn off fetching for a while and then turn it on or disable the instance and enable it, the instance remembers the last run and pulls all events that occurred while it was off. If you don't want this to happen, verify that the instance is enabled and click Reset the "last run" timestamp when editing the instance. Also, note that "last run" is retained when an instance is renamed.

After configuring the instance, you may need to set up a correlation rule to ingest issues.

Correlation rules are predefined logic or patterns that Cortex AgentiX uses to identify relationships between disparate events occurring across an organization's IT environment. If the conditions specified in the rule are met, Cortex AgentiX generates an issue.

How to fetch issues

1. Navigate to Settings → Data Sources & Integrations, find and select the integration, and click Add Instance.

2. In the integration's dialog box, select Fetch issues.

After this setting is enabled, Cortex AgentiX searches for events that occurred within the time frame set for the integration, which is based on the specific integration. The default is 10 minutes, but it can be changed in the integration script.

3. (Optional) In the Issue Fetch Interval field, set the interval of hours and minutes to fetch alerts (default 1 minute).

4. (Optional) If the Issue Fetch Interval field does not appear, add it to the integration.

Relevant for any issue fetching integration:

a. For integrations installed from a content pack, select the duplicate integration button.

If you have already duplicated the integration, click the Edit integration's source button.

b. In the Basic section, select the Fetch issues checkbox.

In the Parameters section, you can see that the `IssueFetchInterval` parameter is added. Change the default value if necessary.

c. Click Save to save the changes.

5. To generate issues, add correlation rules, as required.

**NOTE:**

Some content packs include preconfigured correlation rules, but you should review them to see if they suit your use case and duplicate them if required. Go to Threat Management → Detection Rules → Correlations, search for the relevant rule, right-click, and select Preview Rule. For example, the ServiceNow v2 Alerts (automatically generated) correlation rule uses the following XQL Query:

```
dataset = servicenow_v2_generic_alert_raw
| filter _alert_data != null
| alter alert_severity = json_extract_scalar(_alert_data, "$.severity")
| alter alert_category = json_extract_scalar(_alert_data, "$.alert_category")
| alter alert_name = json_extract_scalar(_alert_data, "$.alert_name")
| alter alert_description = json_extract_scalar(_alert_data, "$.alert_description")
```

You may want to update the query by defining complex, multi-source detection logic or add filters, such as alert severity or assignee.

### 3.2.3.4.1 | Map fields to issue types

Abstract

You can create independent mappers for integrations.

Mappers enable you to map information from incoming events to the issue fields that you have in your system. You can map to system issue fields or custom issue fields.

Mapping event attributes or issue fields takes place in two stages. First you map all of the fields that are common to all issues in the default mapping. Second, you map the additional fields that are specific for each issue indicator type, or overwrite the mapping that you used in the default mapping.

**NOTE:**

In the Classification & Mapping page, the mapping does not indicate for which issue types they are configured. Therefore, when creating a mapper, it is best practice to add to the mapper name, the issue types the mapper is for. For example, Mail Listener - Phishing.

**NOTE:**

When mapping a list, we recommend you map to a multi select field. Short text fields do not support lists. If you do need to map a list to a short text field, add a transformer in the relevant playbook task, to split the data back into a list.

You can use this procedure for creating a classifier or duplicating an existing mapper for issue types.

1. Navigate to Settings → Configurations → Object Setup → Issues → Classification & Mapping.



2. Click New and select Issue Mapper (incoming). The Issue Mapper maps all of the fields you are pulling from the integrations to the issue fields in your layouts.

3. Under Get data, select from where you want to pull the information based on where you want to map the issue types.

- Pull from instance - select an existing integration instance.
- Select schema - when supported by the integration, this pulls all of the fields for the integration from the database. This enables you to see all of the fields for each given event type that the integration supports.
- Upload JSON - upload a formatted JSON file which includes the field you want to map.

4. Under Issue Type, start by mapping out the Common Mapping. This mapping includes the fields that are common to all of the issue types and will save time having to define these fields individually in each issue type.

5. Click the event attribute to which you want to map. You can further manipulate the field using filters and transformers.

You can click Auto Map to automatically map fields with common or similar names to fields in Cortex AgentiX . For example, Severity to Importance or Description to Description.

6. Repeat this process for the other issue types for which this mapping is relevant.

7. Click Save.

8. Go to Settings → Data Sources & Integrations.

- Select the integration instance to which you want to apply the mapper.
- In the integration settings, under Mapper (incoming) select the mapper you created and click Save.

#### 3.2.3.4.2 | Classify events using a classifier for issue types

##### Abstract

Classify events using a classification key in an integration ingestion.

When an integration fetches issues, it populates the rawJSON object in the issue object. The rawJSON object contains all of the attributes for the event. For example, source, when the event was created, the priority that was designated by the integration, etc. When classifying the event, you want to select an attribute that can determine the event type.

You can use this procedure for creating a classifier or duplicating an existing classifier.

1. Go to Settings → Configurations → Object Setup → Issues → Classification & Mapping.

2. Click New and select Issue Classifier.

If you want to duplicate the classifier, select the relevant classifier and then duplicate it.

3. Under Get data, select from where you want to pull the information based on which you will classify the issue types.

- Pull from instance - select an existing integration instance.
- Select schema - when supported by the integration, this will pull all the fields for the integration from the database from which you can select by which to classify the events.
- Upload JSON - upload a formatted JSON file which includes the field by which you want to classify.

4. In the Select Instance field, select the instance from where you want to choose the value.

5. In the Data fetched from select the value by which you want to classify the events.

6. Drag values from the Unmapped Values column to the relevant issue type on the right.

You can optionally choose a default issue type for unclassified issues from Direct unclassified events to: Select.



2 Unmapped Values

Select the field to identify the type of the alert

Result:Malware

Drag classifier values to the alert type on the right.

Malware    Phishing

Direct unclassified events to: [Select](#)

**Alert Types**

- Access
- Authentication
- C2Communication
- Defacement
- DoS
- Exfiltration
- Exploit
- Hello World Alert
- Hunt
- Indicator Feed
- Job
- Lateral Movement

7. Click Save.

8. Go to Settings → Data Sources & Integrations.

a. Select the integration to which you want to apply the classifier.

b. In the integration settings, under Classifier, select the classifier you created and click Save.

#### 3.2.3.5 | Forward Requests to Long-Running Integrations

##### Abstract

Configure and manage long-running integrations to export internal data from Cortex AgentiX.

Some long-running integrations provide internal data via API calls to your third-party software, such as a firewall. You can set up Cortex AgentiX to allow third-party software to access long-running integrations installed either on the Cortex AgentiX tenant or on an engine.

Long-running integrations provide internal data via API calls such as:

Integration	Description	See More
O365 Teams (Using Graph API)	Get authorized access to a user's Teams app in a personal or organizational account.	O365 Teams (Using Graph API)
Generic Webhook	Creates cases on event triggers. The trigger can be any query posted to the integration.	Generic Webhook



Integration	Description	See More
Generic Export Indicators Service	<p>Use the Generic Export Indicators Service integration to provide an endpoint with a list of indicators as a service for the system indicators.</p> <p><b>NOTE:</b></p> <p>This integration replaces the External Dynamic list integration, which is deprecated. For more information about how to set up the integration, see <a href="#">Manage external dynamic lists</a>.</p>	Generic Export Indicators
Microsoft Teams	Send messages and notifications to team members.	Microsoft Teams
TAXII Server	Provides TAXII Services for system indicators (Outbound feed).	TAXII Server
TAXII2 Server	Provides TAXII2 Services for system indicators (outbound feed). You can choose to use TAXII v2.0 or TAXII v2.1.	TAXII2 Server
PingCastle	Listens for PingCastle XML reports.	PingCastle
Publish List	Publishes Cortex AgentiX lists for external consumption.	Publish List
Simple API Proxy	Provides a simple API proxy to restrict privileges or minimize the number of credentials issued at the API.	Simple API Proxy
Syslog v2	Opens cases automatically from Syslog clients.	Syslog v2
Web File Repository	Make your environment ready for testing purposes for your playbooks or automations to download files from a web server.	Web File Repository

**NOTE:**

- When running on the tenant, you can only use long-running integrations provided by Cortex AgentiX, you cannot create custom ones. Custom long-running integrations are supported only on engines at this time.
- Configuring custom certificates or private API Keys in the long-running integration instance is supported only on engines, not on the Cortex AgentiX tenant.
- If you have configured a range of Approved IP Ranges under Allowed Sessions on the Security Settings page, any incoming communication must be from approved IP addresses.

Credentials

For long-running integrations running on a tenant, you must set a username and password. For long-running integrations running on an engine, we strongly recommend setting a username and password, but it is not required.

Users with sufficient permissions can set the username and password for specific integration instances on the Data Sources & Integrations page.

[Test the long-running integration connection](#)



- Integration instance running on a tenant

You can use CURL commands from any terminal to access and test the long-running integration. The string `xdr` in the URL must be replaced by `crtx` and the data URL must always be prefixed by `ext-`.

**NOTE:**

For the TAXII Server and TAXII2 Server integrations, the `xdr` string is automatically replaced by `crtx`. For the Microsoft Teams integration, you can use the `microsoft-teams-create-messaging-endpoint` command to get the correct messaging endpoint based on the server URL, the server version, and the instance configurations. For more information, see Microsoft Teams.

Example:

Tenant URL: `https://crtx-cnt-onr-xsiam-dran-9c0.xdr-qa2-uat.us.com`

Request URL: `https://ext-crtx-cnt-onr-xsiam-dran-9c0.crtx-qa2-uat.us.com/xsoar/instance/execute/edl_instance_01?q=type:ip`

CURL: `curl -v -u user:pass https://ext-crtx-cnt-onr-xsiam-dran-9c0.crtx-qa2-uat.us.com/xsoar/instance/execute/edl_instance_01?q=type:ip`

- Integration instance running on an engine

You can use CURL commands from any terminal to access and test the long-running integration at the engine URL:

`http://<engine-address>:<integration listen port>/`

For example, `curl -v -u user:pass http://<engine_address>:<listen_port>/?n=50`

Curl request parameters

When sending a curl request to the URL, use the following parameters:

Argument	Description	Example
<code>n</code>	The maximum number of entries in the output. If no value is provided, will use the value specified in the List Size parameter in the integration instance settings.	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?n=50</code>
<code>s</code>	The starting entry index from which to export the indicators.	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?s=10&amp;n=50</code>
<code>v</code>	The output format. Supports PAN-OS (text), CSV, JSON, mwg, and proxysg (alias: bluecoat).	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?v=json</code>
<code>q</code>	The query is used to retrieve indicators from the system.	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?q="type:ip and sourceBrand:my_source"</code>
<code>t</code>	Only with mwg format. The type is indicated at the top of the exported list. Supports: string, applcontrol, dimension, category, ip, mediatype, number, and regex.	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?v=mwg&amp;t=ip</code>
<code>sp</code>	If set, will strip ports off URLs; otherwise, will ignore URLs with ports.	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?v=text&amp;sp</code>
<code>di</code>	Only with PAN-OS (text) format. If set, will ignore URLs that are not compliant with PAN-OS URL format instead of being rewritten.	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?v=text&amp;di</code>



Argument	Description	Example
cr	If set, will strip protocols off URLs.	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?v=text&amp;pr</code>
cd	Only with proxysg format. The default category for the exported indicators.	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?v=proxysg&amp;cd=default_category</code>
ca	Only with proxysg format. The categories that will be exported. Indicators not in these categories will be classified as the default category.	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?v=proxysg&amp;ca=category1,category2</code>
tr	Only with PAN-OS (text) format. Whether to collapse IPs. <ul style="list-style-type: none"> <li>• 0 - Do not collapse.</li> <li>• 1 - Collapse to ranges.</li> <li>• 2 - Collapse to CIDRs</li> </ul>	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?q="type:ip and sourceBrand:my_source"&amp;tr=1</code>
tx	Whether to output CSV formats as textual web pages.	<code>https://ext-&lt;tenant-address&gt;/instance/execute/&lt;ExportIndicators_instance_name&gt;?v=csv&amp;tx</code>

Define a listening port for long-running integrations

When configuring a long-running integration instance, you may need to define a listening port.

- **Integration Instance Running on a Tenant**

If the long-running integration runs on the Cortex AgentiX tenant, you do not need to enter a Listen Port in the instance settings. The system auto-selects an unused port for the long-running integration when the instance is saved.

- **Integration Instance Running on an Engine**

You must set the Listen Port for access when configuring a long-running integration instance on an engine. Use a unique port for each long-running integration instance. Do not use the same port for multiple instances.

### 3.2.3.6 | Manage credentials

Credentials simplify and compartmentalize administrative tasks, and enable you to save login information without exposing usernames, passwords, certificates, and SSH keys. You can reuse credentials across multiple systems, for example, when using the same administrator password across multiple endpoints.

After you set up a credential, you can configure integration instances to use it instead of entering the name and password manually.

How to add credentials to an integration instance

1. Create the credential.

a. Select Settings → Configurations → Integrations → Credentials → New Credential.

b. Add the following parameters:

Parameters	Description
Credential Name	The name of the credential. You select this name when adding the credential to the integration instance.



Parameters	Description
Username	The username for the credential.
Workgroup	The workgroup to associate this credential with. Relevant for third-party services, such as Active Directory, CyberArk, and HashiCorps.
Password	The password for the credential. For example, add the API Key when defining the API credential.
Certificate	Certificate or SSH to use for the credential.

- c. Save the credential.
2. Add the credential to the integration instance.
- a. Go to Settings → Data Sources & Integrations and select the integration.
  - b. Click Add Instance.
  - c. Locate the relevant section and click Switch to credentials.  
If there is more than one credential, select the relevant credential.
  - d. Test and Save & Exit the integration instance.

#### 3.2.4 | Troubleshoot Integrations

##### Abstract

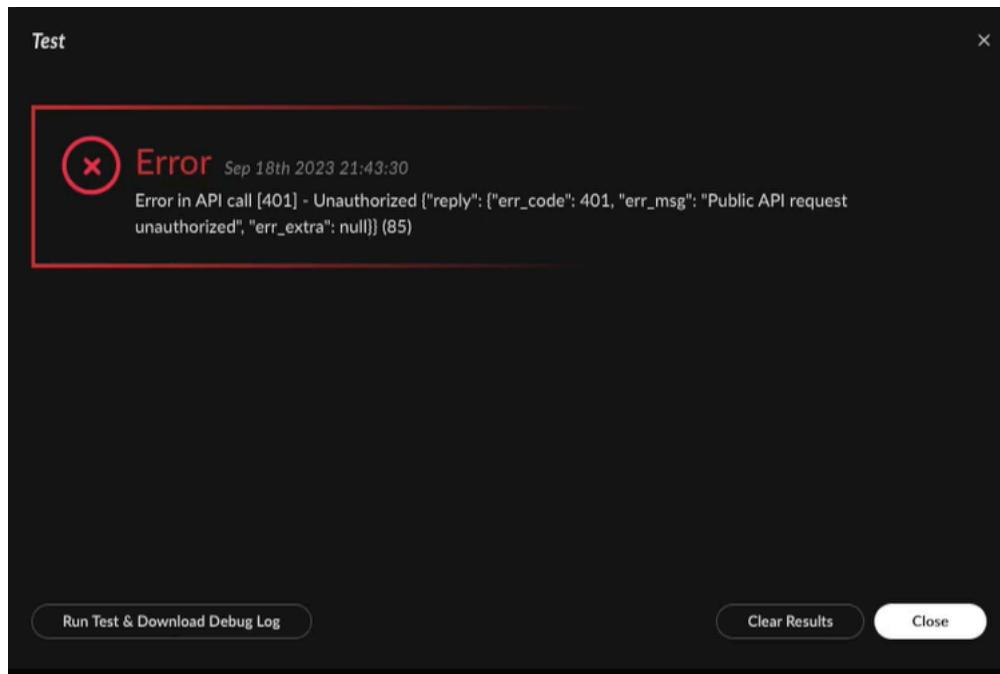
Learn how to troubleshoot your integration in Cortex AgentiX.

The Troubleshooting Instances dashboard provides you with insight into command execution errors. When troubleshooting integrations, we recommend the following steps:



- Use the Test button in the integration instance.
- Verify the integration settings. Check settings such as usernames, URLs, and passwords.
- Download the debug log file and review its contents.

In the following example, you receive a 401 unauthorized error code after testing the integration.



Click Run Test & Download Debug Log, to download the debug file locally. You can verify what server the URL request is being forwarded to and any other reasons as to why you received this error code. The 401 unauthorized error code usually relates to invalid error credentials, expired tokens, or incorrect API settings.

- Enable verbose or debug-level logging on the integration.

If you are unable to fix the integration, contact Customer Support for further assistance.

### 3.2.5 | Measuring data freshness

#### Abstract

Learn more about the data freshness metrics collected by Cortex AgentiX.

Cortex AgentiX provides metrics that calculate the freshness of your ingested data and highlight delays in your data collection. The metrics calculate the freshness delay value by measuring the difference between log creation at the source (`_TIME`) and ingestion into Cortex AgentiX (`_INSERT_TIME`).

Metrics are collected and calculated per data source during five-minute aggregation periods and allocated into the following buckets. The recorded freshness delay value is the top value in the range of the bucket:

- 0 to 30 seconds â 30 seconds
- 30 to 60 seconds â 60 seconds
- 60 seconds to 5 minutes â 300 seconds
- 5 minutes to 1 hour â 3,600 seconds
- 1 hour to 24 hoursâ 86,400 seconds
- 24 hours to weekâ 604,800 seconds

Metric	Description
data_freshness_max_delay	Maximum freshness delay value among all log entries in an aggregation period. This reflects the worst case.



Metric	Description
data_freshness_median	Median freshness delay value among all log entries in an aggregation period. 50% of values are smaller than the median, and 50% of values are higher or equal to the median.
data_freshness_ninetieth_percentile	Ninetieth percentile of delay values among all log entries in an aggregation period. This delay value is 90% higher than other log entry differences. It reflects the worst case, but eliminates the spikes.

The metrics are saved to the `metrics_source` dataset and are also available in the `metrics_view` preset.

#### NOTE:

- The `max_delay` metric is taken from the maximum bucket value with a restricted limit; therefore, metrics show whole numbers.
- The `median` and `ninetieth_percentile` metrics are statistical calculations that give an approximation of the real value; therefore, metrics show decimal numbers.
- Time slots with a zero log count or zero byte count display records with zero values. Subsequently, the data freshness metrics will also have zero values.
- Timezone differences between `_TIME` and `_INSERT_TIME` might cause time skews with negative differences. Negative differences are rounded to zero values.

### 3.2.6 | Overview of data ingestion metrics

#### Abstract

Learn more about the data ingestion health metrics in the `metrics_source` dataset and the `metrics_view` preset.

#### PREREQUISITE:

For Cortex AgentIX to monitor data ingestion health and create health issues, you must enable Cortex - Analytics. Go to Configurations → Cortex - Analytics. For more information, see [Enable the Analytics Engine and Identity Analytics](#).

The data ingestion metrics are calculated in 5-minute aggregation periods and saved to the `metrics_source` dataset and `metrics_view` preset. These metrics measure the amount, size, and rate at which logs are ingested by a data source:

Metric	Description
<code>total_size_bytes</code>	Total size (in bytes) of the logs collected during the aggregation period.
<code>total_size_rate</code>	Average size (in bytes per second) of the logs collected during the aggregation period.
<code>total_event_count</code>	Total number of logs collected during the aggregation period
<code>total_event_rate</code>	Average number (in count per second) of logs collected during the aggregation period.

In the `metrics_source` dataset, the data ingestion metrics are saved alongside additional fields that describe the data source associated with the metrics. Only entries with ingestion metric values greater than zero are saved in the dataset. Entries with zero values are not saved in this dataset.

`metrics_view` is a preset for data in the `metrics_source` dataset. The preset also simulates completion of entries with zero values in data ingestion metrics at runtime, which allows effective use of metrics. Therefore, when investigating disruptions in data collection, we recommend using the `metrics_view` preset in XQL queries and correlation rules.

Cortex AgentIX built-in data ingestion monitoring and issue mechanism uses the data ingestion metrics to identify disruptions in the data ingestion pipeline. Using analytical logic, Cortex AgentIX creates an ingestion baseline for each data source that reflects the routine pattern of log collection. If a data source isn't ingesting logs, or there is a significant deviation from the baseline, ingestion issues are triggered. You can see all ingestion issues on the Health Issues page. To troubleshoot or investigate an issue, right-click an issue and click Investigate in XQL query. For more information, see [Investigate and resolve health issues](#).

In addition, you can create your own custom logic for data ingestion health monitoring by setting up correlation rules that monitor the data ingestion metrics. For more information, see [Creating correlation rules to monitor data ingestion health](#).



The following table describes all the fields in the `metrics_source` dataset and `metrics_view` preset:

[Read more...](#)

Field	Type	Description
total_size_bytes	Integer	Total size (in bytes) of the logs collected during the aggregation period.
total_size_rate	Integer	Average size (in bytes per second) of the logs collected during the aggregation period.
total_event_count	Integer	Total number of logs collected during the aggregation period.
total_event_rate	Integer	Average number (in count per second) of logs collected during the aggregation period.
data_freshness_max_delay	Float	Maximum delay value from all log entries in a record between log creation at the source and ingestion into Cortex AgentiX (in seconds).
data_freshness_median	Float	Median delay value from all log entries in a record between log creation at the source and ingestion into Cortex AgentiX (in seconds).
data_freshness_ninetieth_percentile	Float	Ninetieth percentile of delay values from all log entries in a record between log creation at the source and ingestion into Cortex AgentiX (in seconds).
last_seen	Datetime	Time that the last logs were collected.
_vendor	String	Vendor of the observing data source.
_product	String	Product name of the observing data source.
_device_id	String	(For firewall devices) Device ID
_log_type	String	(For firewall devices) Log type
_collector_name	String	(Event Metadata) Name of the collector instance.
_collector_id	String	(Event Metadata) ID of the XDR Collector.
_collector_ip	String	(Event Metadata) IP address of the XDR Collector.
_reporting_device_name	String	(Event Metadata) Host name of the device where the log originated.
_reporting_device_ip	String	(Event Metadata) IP Address of the device where the log originated.
_final_reporting_device_name	String	(Event Metadata) Hostname of the device that the log was extracted from.



Field	Type	Description
_final_reporting_device_ip	String	(Event Metadata) IP of the device that the log was extracted from.
_time	Datetime	Timestamp of the interval.
_insert_timestamp	Datetime	Recorded time of the entry.

### 3.2.7 | Verify collector connectivity

#### Abstract

Verify collector connectivity and troubleshoot collector errors.

You can verify the connectivity status of a collector instance on the Data Sources & Integrations page. Instances are grouped by integration, and a status icon shows a summary of instance statuses for each integration. Expand the integration section to see the status of each individual instance, and hover over the status icons to see details about warning or error statuses.

In addition, Cortex AgentiX creates Collection health issues if connectivity disruptions occur in your collection integrations, custom collectors, and Marketplace integrations. For more information, see About health issues.

#### Troubleshooting collector errors

Where can I see if I have a connectivity error on a collector instance?

On the Data Sources & Integrations page, instances in error status display an error icon. Hover over the error icon next to the instance name to see the error message as received from the API.

Where can I trace the connectivity changes of a collector instance?

Each status change of an instance is logged in the `collection_auditing` dataset. Querying this dataset can help you see all the connectivity changes of an instance over time, the escalation or recovery of the connectivity status, and the error, warning, and informational messages related to status changes.

Example 7.

This example searches for status changes on Strata IOT integrations:

```
dataset = collection_auditing
|filter collector_type = "STRATA_IOT"
```

How can I set up correlation rules to trigger collection issues?

Cortex AgentiX provides OOTB Collection issues that are triggered when a data collector instance is in error status, which means it is disconnected or not sending data. In addition, you can set up your own correlation rules that trigger collection issues for your specific needs. For example, you might want to be notified if a high-profile collector is in warning status so that you can fix the problem and prevent the collector from disconnecting.

Example 8. Example: Trigger collection issues for warning statuses on the STRATA\_IOT collector

In this example, a correlation rule triggers a Collection issue if an integration of the Strata IOT collector changes to warning status. Any issues will appear on the Health Issues page.

Example XQL:

```
dataset = collection_auditing
|filter classification = "Warning" and collector_type = "STRATA_IOT"
```

Additional fields to specify in the correlation rule:

Field	Value
Time Schedule	Hourly
Query time frame	1 Hour



Field	Value
Issue Suppression	Select Enable issue suppression.
Action	Select Generate issue.
Issue Domain	Health
Severity	Medium
Category	Collection <b>NOTE:</b> If an issue is triggered, the investigation options in the right-click menu of the Health Issues pages are context-specific. Make sure that you specify the relevant issue category.

### 3.2.8 | Creating correlation rules to monitor data ingestion health

#### Abstract

See examples of correlation rules for monitoring data ingestion health.

In addition to the OOTB Ingestion health issues, you can build your monitoring logic for ingestion by creating correlation rules that are specific to your requirements. You can create rules that monitor the data ingestion metrics for a specific source within a specific timeframe, and trigger ingestion health issues if there is a deviation from the regular pattern of log collection.

The following examples can help you set up your own correlation rules with the data ingestion metrics:

Example 1: No logs collected from a data source for 1 hour

In this example, the correlation runs every hour and calculates the number of logs that are collected for each data source over the previous hour. If no logs are collected for a data source during an aggregation period, a security issue is triggered.

Example XQL:

```
preset = metrics_view
| comp sum(total_event_count) as total_event_count_sum by _collector_id, _collector_ip,
_collector_name, _collector_type, _final_reporting_device_ip, _final_reporting_device_name,
_broker_device_id, _vendor, _product
| filter total_event_count_sum = 0
```

Addition fields to specify in the correlation rule:

Field	Value
Time Schedule	Hourly
Query time frame	1 Hour
Issue Suppression	Select Enable issue suppression.
Fields	Uncheck <code>total_event_rate_sum</code> , leave other fields checked.
Action	Select Generate issue.



Field	Value
Issue Domain	Health
Severity	High
Type	Ingestion
Issue Fields Mapping	Select Use preconfigured fields to map the fields that are relevant to data ingestion health.

Example 2: No logs received from a Firewall for 20 minutes

In this example, the correlation runs every 20 minutes and calculates the number of logs that are received for each firewall in a lookup dataset during the last 20 minutes. If no logs are received from a device during an aggregation period, a security issue is triggered.

Example XQL:

```

preset = metrics_view
| join conflict_strategy = left type = inner (dataset = ngfw_device_Id_keepalive
| fields _device_id) as devices devices._device_id = _device_id | comp sum(total_event_count)
as total_event_count_sum by _device_id, _product, _vendor
| filter total_event_count_sum = 0

```

Addition fields to specify in the correlation rule:

Field	Value
Time Schedule	Every 20 minutes
Query time frame	20 minutes
Issue Suppression	Select Enable issue suppression.
Fields	Uncheck <code>total_event_rate_sum</code> , leave other fields checked.
Action	Select Generate issue.
Issue Domain	Health
Severity	High
Type	Collection
Issues Fields Mapping	Select Use preconfigured fields to map the fields that are relevant to data ingestion health.

### 3.3 | Configure the Cortex Agentic Assistant

Abstract

Create and manage agents and actions in the Agents Hub and configure access to the Cortex Agentic Assistant.

Create and manage agents and actions in the Agents Hub, configure access to the Cortex Agentic Assistant to create an AI agent workforce.



### 3.3.1 | Agentic Assistant components and concepts

#### Abstract

Learn about the key components and concepts, such as agents and actions in the Cortex Agentic Assistant

The Cortex Agentic Assistant uses the following components and concepts:

Name	Description
Actions	Actions wrap diverse capabilities (such as playbooks, scripts, and commands) to make them accessible and executable by an agent. You can use out-of-the-box system actions or register new actions.
Agent	An agent is a virtual persona that creates and executes domain-specific plans, at your request, to assist in your day-to-day SOC operations. An agent has roles and permissions that provide guardrails. Each agent is assigned a collection of actions that it can use as part of plans.  The agent chooses the most relevant actions to fulfill a user's request. Agents process user requests, create plans, and orchestrate actions based on their goals and permissions (RBAC and SBAC).  You can use the following types of agents: <ul style="list-style-type: none"><li>System agents that are provided by Cortex AgentiX for specific use cases.</li><li>Custom agents that users have created.</li></ul> Some agents provide relevant chat conversation starters under the chat prompt. For examples of conversation starters, see Agentic Assistant use cases.  <b>NOTE:</b> Agents are bound by the same rules and robust permissions as a human user. In addition, you can mark actions that make real-world changes in production systems as sensitive, requiring a quick manual review and confirmation, ensuring peace of mind before critical system changes are made.
Plan	A sequence of actions that run in parallel or sequentially to satisfy a user request. The agent dynamically chooses relevant actions to resolve the prompt.
Conversation	A sequence of user requests that maintains context across interactions.
Request	A user request from the agent with an end goal, triggering a plan.

### 3.3.2 | Agents Hub

#### Abstract

Learn about personal and system agents in the Agents Hub

In Cortex AgentiX, you can interact with agents in the Agentic Assistant chat to automate case and issue investigation and response. Agents create and execute plans, which are sequences of actions (such as playbooks, scripts, and commands) designed to fulfill your requests.

Actions and agents are managed in the Agents Hub. To open the Agents Hub, click on the agent chat icon  in the upper right hand corner, click the side panel icon  to expand the menu if needed, and then click the Agents Hub menu item.



#### NOTE:

To manage agents in the Agents Hub, you must have the proper permissions. For more information, see Agentic Assistant role-based access control.



All Agents (87)

Sort by Creation time

Threat Intel By Cortex

Gathers fresh threat data, enriches indicators and vulnerabilities, links them to past or current incidents, and...

22 Actions

Network Security By Cortex

Manages Palo Alto Networks Panorama and Firewalls, as well as third-party network security products. Streamlines work...

26 Actions

IT By Cortex

Automates identity lifecycle enforcement, real-time containment on endpoints and networks, vulnerability and...

22 Actions

Endpoint Investigation By Cortex

Unifies host-level containment, forensic collection, and remediation across all major EDR/XDR platforms while...

29 Actions

The Agents Hub includes the following components:

- **Actions**

Actions wrap diverse content items (such as playbooks, scripts, AI prompts, and commands) to make them accessible and executable by an agent. Cortex AgentiX provides system actions, and you can also create your own actions. Custom actions can be created from scripts, commands, and AI prompts.

You can register new actions through the Agents Hub or from the Scripts or AI Prompts page. Actions can include functionality such as sending emails, extracting data, enriching information, or opening support cases. Multiple actions can be created from a single script, command, or AI prompt, if needed. An action can be added to multiple agents.

- **Agents**

An agent is a virtual persona that creates and executes domain-specific plans, at your request, to assist in your day-to-day SOC operations. An agent has roles and permissions that provide guardrails. Each agent is assigned a collection of actions that it can use as part of plans.

The agent chooses the most relevant actions to fulfill a user's request. Agents process user requests, create plans, and orchestrate actions based on the user's goals and permissions (RBAC and SBAC).

Cortex AgentiX provides system agents, and you can also create custom agents. In the Agentic Assistant chat, you can select any system agent, any agent you created, or any public agent.

- From the Agents tab of the Agents Hub, you can hover over any agent card to see the View option. Click View to see all the actions assigned to the agent and their status.

In the Agents Hub, you can do the following:

- Register scripts, commands, and AI prompts as custom actions. After a script, command, or AI prompt is registered as a custom action, it can be assigned to agents and used in plans. For more information, see [Manage actions](#).
- View system actions and edit existing custom actions.
- Build agents and assign actions to agents.
- Enable and disable system agents. System agents have access to system actions that are assigned to the agent.
- Start a chat with any agent, by clicking the more options icon on the agent card and clicking Start chat.



### 3.3.2.1 | Manage actions

#### Abstract

Manage actions that can be used by agents.

Actions wrap diverse capabilities (such as playbooks, scripts, AI prompts, and commands) to make them accessible and executable by an agent. You can use out-of-the-box system actions or register new actions.

#### NOTE:

To manage actions in the Agents Hub, you must have the correct permissions. For more information, see [Agentic Assistant role-based access control](#).

There are two types of actions in the Agents Hub:

- **System actions:** Cortex AgentiX contains more than 50 out-of-the-box system actions that can be disabled or enabled, but cannot be edited or deleted.  
To find and install additional content packs that include actions, go to Marketplace and select Content pack includes and Actions.
- **Custom actions:** Users can register existing or new scripts, commands, and AI prompts as actions. Custom actions can be edited, deleted, enabled, or disabled.

Any action marked as sensitive to require user approval requires explicit user approval before execution. This is particularly crucial for operations that might alter system reality or affect an organization's budget, such as isolating an endpoint or revoking user access. System actions are marked sensitive if they affect system reality. When creating custom actions, you decide which actions should be marked as sensitive for your organization.

#### Manage existing actions

From the Actions tab of the Agents Hub, click  for an action to edit, delete, or disable an existing custom action. System actions can only be enabled or disabled.

#### Search, filter, and sort actions

You can use the dropdown filter to search all actions, custom actions, system actions, enabled actions, or disabled actions.

You can sort actions by most used, creation time, or update time.

### 3.3.2.2 | Register actions

#### Abstract

Register custom actions that can be used by agents.

You can register scripts, commands, and AI prompts as actions in the Agents Hub. After a script, command, or AI prompt is registered as an action, it can be added to one or more agents. The agents can then execute the action as part of plans.

#### NOTE:

To register actions, ensure you have the correct permissions. For more information, see [Agentic Assistant role-based access control](#).

When you register an action, you provide a description, goal, and, optionally, a few-shot examples. This information helps agents understand how the action should be used.

When registering or editing an action, you can choose which specific inputs and outputs are visible to the LLM. For example, a script might have two inputs and five outputs, but for this action, only one input and two outputs are required, and only those are included in the action. This helps to create more focused actions and reduces unnecessary complexity.

#### IMPORTANT:

A single content item can be registered as different actions, with each action using different inputs and outputs from the same script. Only register the same script, command, or AI prompt as a new action if it is required for your use case, as providing an agent with many actions with overlapping abilities can reduce the ability of the agent to choose the most appropriate action.

While you can create multiple actions from a single content item, each action must have a different name.

If you try to register a script, command, or AI prompt that is already registered, you are presented with a list of the actions already using it, and you can review and decide if any of them are relevant for your current use case. If not, you can register the script, command, or AI prompt again as a different action.

How to register an action

1. Do one of the following:



- Click the Agentic Assistant icon in the upper right hand corner and expand the side panel  to access the Agents Hub menu item. From the Actions tab of the Agents Hub, click Register new action.
    - Within the script creation or editing screen, click  when viewing or editing a script and select Register as action.
    - Within the AI Prompts library, select a prompt and click the more options icon to Register as action.
2. If you clicked Register as action from the Scripts or AI Prompts page, the name is prepopulated in the Content chosen field. If you clicked Register action from the Agents Hub, select script, command, or AI prompt for the Type of content and select the content you want to register.
3. Enter an action Name, a short description of what the action does. Example: `Extract_email`.
4. Describe the Goal of the action.
5. (Optional) By default, Mark action as sensitive to require user approval is selected, and the agent prompts the user to approve before executing the action. If you do not want the action marked as sensitive, clear the checkbox.
6. (Optional) Provide Few-shot examples to help the agent understand the context and the appropriate situations to invoke a specific action.
7. Click Next.
8. Choose your Action Parameters:

**NOTE:**

Mandatory arguments cannot be deselected.

- Choose which arguments and inputs to include in the action. If a content item contains descriptions of the inputs, the descriptions are prepopulated. If not, you can provide short descriptions. The descriptions help the agent understand the purpose of each input.
- (Optional) Enter a default value for each input. The default value is used when the user does not specify the input.
- Choose which script outputs to include in the action. If the content item contains descriptions of the outputs, the descriptions are prepopulated. If not, you can provide short descriptions. The descriptions help the agent understand the purpose of each output.

9. Save changes.

### 3.3.2.3 | Manage agents

#### Abstract

Edit, disable, or delete existing agents.

Agents create and execute step-by-step plans dynamically, choosing relevant actions based on a user's request. Each agent has a model, a user context, a conversation context, and a set of actions that it can perform. Users engage with agents through conversations in the chat interface.

Permissions for the Agentic Assistant and the Agents Hub can be found under CORTEX AGENTIC ASSISTANT in the role permissions when creating or edit a role. For more information, see Agentic Assistant role-based access control

There are two types of agents in the Cortex Agentic Assistant:

- Custom agents:** Each user can create one or more agents that have the same or fewer permissions as the user, ensuring agents operate with the least necessary privileges required. These permissions automatically update if the user's roles or permissions change. When users create custom agents, they can create a private agent only they can access, or a public agent all users can access.
- System agents:** System agents come out-of-the-box and are not linked to a specific user; instead, they possess their own defined roles and permissions. A system agent may include actions that the user does not have permission to execute. All users have access to all system agents, but plan execution is limited by the permissions of the individual user.

#### Agent management

You can edit, delete, disable, or enable custom agents by clicking the more options  icon for the agent.

You can edit, enable, or disable system agents by clicking the more options  for the agent. The edit option for system agents is limited to adding specific instructions for the agent such as tone, style, format, and priorities.

You can click on an Agent to view all actions assigned to the agent. There are three possible statuses for actions assigned to an agent:



- **Enabled** (green circle with a check mark): The action is enabled and available for the agent to use.
- **Disabled** (grey circle with an x): The action has been disabled and is not available for the agent to use.
- **Unavailable content** (grey circle with a horizontal line): The content the action is based on is not available. To use the action, the content item must be installed and configured.

**NOTE:**

In some cases, an agent may include actions with content items that are not relevant for all licenses. If that occurs, the grey circle appears, but you are not able to install the related content.

#### Search, filter, and sort existing agents

You can use the dropdown filter to search all agents, custom agents, enabled agents, or disabled agents.

You can sort agents by most used, creation time, or update time.

#### 3.3.2.4 | Build agents

##### Abstract

Build new agents.

You can build custom agents in Cortex AgentiX to execute plans and assist in investigations. Custom agents have the same or fewer permissions as the user who creates them. For example, you might want to create an agent with all of your permissions to use for certain investigations, but also create a read-only agent that provides you with information, but does not execute actions on real-world systems. You can create custom agents that are private or that are shared for all users.

When you build an agent, it should contain all actions that you require for your workflow. Agents are self-contained and cannot communicate with other agents or access actions that are not assigned to the agent.

**NOTE:**

To build agents in the Agents Hub, you must have view/edit permissions. For more information, see [AgentiX Assistant role-based access control](#).



1. Click on the agent chat icon in the upper right hand corner, click the side panel icon to expand the menu if needed, and then click the Agents Hub menu item.
2. From the Agents tab of the Agents Hub, click Create agent.
3. Complete the following agent detail fields:

Field	Description	Required
Agent Name	A short description name for the agent. Each agent must have a different name.	Yes
Color	The color for the icon that appears in the agent list.	No
Description	A description of the agent's purpose or area.	Yes



Field	Description	Required
Specific Instructions	<p>Provide the agent with detailed customized instructions. You can include a wide range of directives, from describing the agent's role and preferred terminology to step-by-step processes and structure of the output.</p> <ul style="list-style-type: none"> <li><b>Role:</b> What the agent is supposed to be or act as. Defines its identity and primary function. Example A: SOC tier 1 analyst. As a tier 1 analyst you are responsible for triaging alerts and concluding if an alert is a true or false positive.</li> <li><b>Instructions:</b> The specific rules and behavioral guidelines that tell the agent how to operate and respond. Example: Follow the NIST framework, provide clear and concise recommendations, use critical thinking when conducting analysis.</li> <li><b>Structure:</b> How the agent should format and organize its responses. Examples of possible formats: JSON, Markdown, Array, enum.</li> </ul>	No
Agent access	Choose whether to make the agent a Public Agent. Public agents can be accessed by all users with View/Edit permissions to Interact with Agents. By default, custom agents are only available for the users who created them.	No
Conversation starters	Include up to four prompts that appear under the prompt bar when the user interacts with the agent. Conversation starters help users understand what the agent can do and how to initiate a request.	No

4. Click Next to proceed to the Access Control page.

5. Define which roles and actions the agent can access. To save an agent, there must be at least one role or action selected.

**NOTE:**

If you clear the checkbox for a role, all actions associated with that role are also cleared. The exception is if another role is also selected, which is associated with the same actions.

If you clear the checkbox for an action, all roles associated with that action are cleared. For example, if you select the Investigator role, and Send Mail and Tavily Extract are both actions associated with that role, clearing the check box for Investigator also clears the check box for Send Mail and Tavily Extract. If you then reselect the Send Mail action, the Investigator role is not automatically selected.

Not all actions are associated with a role.

For an agent to be able to run XQL queries, you must add the Cortex - Run XQL Query action. This action is included by default for all system agents.

6. If needed, register one or more new actions by clicking New Action and following the steps in Manage actions.



7. Save Agent.

### 3.3.2.5 | Expand agent capabilities with MCP integrations

#### Abstract

Learn how Agentic agents can leverage tools on third-party MCP servers.

Cortex Agentic Assistant supports native interaction with external environments via the Model Context Protocol (MCP). Agentic Assistant agents can use tools from third-party MCP servers to retrieve data and perform tasks in external systems. For example, an agent can open a Jira issue or check GitHub to see if security scans in a workflow are being bypassed..

The Cortex Agentic Assistant connects to external MCP servers using streamable HTTP and supports both OAuth-based and Authless servers. The MCP server must be accessible via a URL. To communicate with third-party MCP servers, you install the relevant content pack from Marketplace and configure an integration instance. The integration connects to the third-party MCP server to discover available tools and automatically generate agentic actions.

#### MCP integrations

To find MCP content packs, filter for MCP under Types in Marketplace. Examples of MCP content packs include Cloudflare MCP, GitHub MCP, and Atlassian Cloud MCP. You can also use the Generic MCP content pack to connect to MCP servers that do not have their own specific content pack. Each MCP integration includes instructions for providing the required parameters, such as the URL and authentication details.

You can create multiple integration instances for each MCP integration. For example, you might configure one instance of the GitHubMCP integration to connect to a environment with read tools and another instance to connect to an environment with both read and write tools. In addition, the GenericMCP integration can be used to connect to multiple MCP servers, each with a separate integration instance.

When you Test the integration instance, Cortex AgentiX verifies server connectivity.

#### NOTE:

If you are using OAuth-based authentication, the Test button returns an error containing the command to run in the playground in order to test the connection.

All configured MCP integration instances can be viewed in the Settings â Data Sources & Integrations page. You can view each integration instance and verify the status of the connection, Test the connection, enable or disable the integration instance, and view the last discovery timestamp.

#### MCP tool actions

The integration instance checks hourly for new or changed tools exposed by the third-party MCP server. The same checks are also performed every time an integration instance is saved. All discovered tools are automatically registered as AI actions, with the type MCP Tool. Actions are created once per MCP integration instance. If you have multiple integration instances for the same MCP server, multiple actions are created for the same tools. The server name, the tool name, and the name of the integration instance are all included in the name of the action. Actions created through tool discovery are system actions. The actions cannot be edited, but you can enable and disable them and also select or clear the checkbox to mark the action as a sensitive action that requires manual approval. By default, all actions registered from MCP servers are marked as sensitive.

#### NOTE:

- If an MCP tool is removed from the MCP server, the action will be unavailable due to missing content. If the tool is restored on the MCP server, the action is automatically reenabled.
- If you have a development and a production tenant, you can push actions created from MCP tools from development to production.

#### Agents and permissions

To use MCP tool actions, they must be added to custom agents in the Agents Hub. By default, all users with access to the custom agent can use all of the available tools. To restrict access to MCP tools, go to Settings â Configurations â Data Collection â Integration Permissions. You can restrict access for MCP integration instance commands to one or more roles. If you restrict access, only users in the permitted roles can use these actions.

### 3.3.3 | Agentic Assistant role-based access control

#### Abstract

Configure permissions to access Cortex Agentic Assistant features.

Instance and Account admins have full control over the permissions and access that users have to the Cortex Agentic Assistant. Cortex AgentiX uses role-based access control (RBAC) to manage access to the chat, as well as access to view, create, edit, delete, disable, and enable Agents and Actions in the Agents Hub.

By default, Instance and Account admins have full view/edit permissions enabled. When editing or creating other roles, in the Cortex Agentic Assistant â Agents section, you can select the following:



Permission	Description
View/Edit	<p>When selected (and nothing else is checked in this section), the user role can only see actions and public agents in the Agents Hub, but cannot interact with agents.</p> <p>You can also select the following permissions:</p> <ul style="list-style-type: none"> <li>Interact with agents: Users can trigger Agents in the Cortex Agentic Assistant. Users can access their own agents, public agents and system agents.</li> <li>Manage actions: Users can view, create, update, and delete actions.</li> <li>Manage agents: Users can view, create, update, and delete their own custom agents.</li> <li>Agents admin: Users can view, create, update, and delete all actions and agents. Users can enable or disable system actions and agents.</li> </ul>
View	N/A
None	The user role does not see any agents and can't use the chat. The Agents Hub is not visible to the user. Cortex Agentic Assistant is only available for navigation and insights.

## 3.4 | Cortex MCP server

Learn how to install, configure, and use the Cortex MCP Server with Cortex AgentiX.

### 3.4.1 | Cortex MCP server overview

#### Abstract

The Cortex MCP server enables you to leverage Cortex's powerful capabilities directly through natural language. Use built-in tools to manage cases and issues and conduct investigations, with the flexibility to create and customize new tools to fit specific use cases and workflows.

The Cortex MCP Server enables you to access Cortex's powerful features directly within your Large Language Model (LLM) apps. Built on the Model Context Protocol (MCP), a standard for connecting AI models to work with other applications and tools, enabling you to query your Cortex tenant and conduct investigations using natural language.

#### NOTE:

This feature is in **Beta**.

#### Key capabilities

- Investigate
- Use the built-in tools to manage cases and issues, and conduct investigations.
- Customize
- Create, customize, and fine-tune tools to fit specific use cases and workflows.
- Flexible client

The Cortex MCP Server is provided as a downloadable file that can be installed on a local machine or a container. While these instructions use Claude Desktop as the MCP client, you can use any client that supports MCP. More detailed setup instructions are provided in a README file included in the download.

#### NOTE:

The Cortex MCP Server empowers you to integrate AI into your security workflows using natural language. When using LLM-based suggestions, always review and approve actions suggested by the AI before they're executed. We recommend deploying the Cortex MCP server in a secure environment where access is limited to authorized users.

To install, configure, and use the Cortex MCP server:

- Install the Cortex MCP server
- Configure the MCP client
- (Optional) Create custom Cortex MCP server tools



#### 4. Use the Cortex MCP server

##### 3.4.1.1 | Install the Cortex MCP server

###### Abstract

Download, install, and configure the MCP server on your local machine or a container.

With the Cortex MCP Server, you can use natural language in your MCP client to investigate and manage cases and issues. The MCP Server can be run within a Docker container or a Poetry virtual environment. .

This documentation contains instructions for configuring and using the Cortex MCP server. More detailed setup instructions are provided in a README file included in the download.

These instructions use Claude Desktop, but you can use any client that supports MCP.

###### **PREREQUISITE:**

If you are running the Cortex MCP server in a Poetry virtual environment, you must have Python 3.13 or higher.

If you plan to run the Cortex MCP server in a Docker container, you must have Docker installed.

Step 1: Create an API key

###### **NOTE:**

The MCP Server uses public APIs to communicate and is limited by the license quotas available in your tenant. This is particularly relevant when running XQL queries. For more information on running XQL query APIs, see Run XQL query APIs.

1. Select Settings → Configurations → Integrations → API Keys → New Key.

2. In the Role tab, perform for the following:

a. Under Security Level, select Standard.

b. Under Role, select the desired level of access for this key. You can select from predefined roles or custom roles. Roles are available according to what was defined in either the Cortex Gateway or the tenant's Access Management. You can view the configuration of the role selected by expanding the sections under Components.

###### **NOTE:**

It is critical to avoid assigning excessive permissions when creating an API key for the Cortex MCP Server. Since the key has both read and write capabilities, overly broad permissions can lead to unintended actions and potentially compromise your environment. Ensure the key follows the principle of least privilege and is granted only the minimum required access.

c. (Optional) Under Comment, provide a comment that describes the purpose of the API key.

d. (Optional) If you want to define a time limit on the API key authentication, select Enable Expiration Date, and select the expiration date and time. You can track the expiration date of each API key in the API Keys page. In addition, a API Key Expiration notification appears in the Notification Center one week and one day prior to the defined expiration date.

3. (Optional) If Scope-Based Access Control (SBAC) is enabled for the tenant, click Scope, and under Scope Definition, select the scope areas that you want to limit the user role to access for this API.

4. Click Generate to generate the API key.

5. Copy the generated API key and click Done.

###### **IMPORTANT:**

To configure the Cortex MCP Server, you need the Cortex API URL, Cortex API key, and Cortex API key ID. You will not be able to view the API key again after you complete this step. Ensure that you copy the API key before closing the notification.

Step 2: Download and install the Cortex MCP server

1. Go to Settings → Configurations → Integrations → Cortex MCP Server.

2. Download MCP File

3. (Optional) Download the checksum file and run a command such as `shasum` (Linux/macOS) or `certutil` (Windows) to verify the integrity and authenticity of the file. For example: `shasum -a 256 -c cortex-checksum.zip.sha256`.

4. Extract the .zip file.

5. Follow the detailed instructions in the README.md file located in the top directory. Instructions are provided for both Docker and Poetry and include the following:

Docker



- Create an .env file with the environment variables.

**NOTE:**

When using Docker, we recommend using an .env file to set the Cortex API credentials as environment variables. While the credentials can be provided in the MCP client configuration settings, the .env file provides safer handling of API credentials and makes your configuration easily reproducible.

- Build and run the Docker container.

Poetry

- Install Poetry.
- Create and activate a virtual environment.
- Install project dependencies.
- Provide the required variables in the Python runtime environment.

Step 3: Run the Docker container or start the server in the Poetry virtual environment

**NOTE:**

By default, stdio (standard input/output) is used. You can also configure Streamable HTTP, to send requests directly to the tenant instead of through the MCP client. Streamable HTTP can be useful for testing in the browser without a MCP client and to bypass limits that may be in place for your MCP client. For Docker, you can include the Streamable HTTP variables in the .env file. You can also include it as a flag when you start the server in the Python virtual environment.

Docker

```
docker run --env-file .env -it cortex-mcp
```

Poetry virtual environment

```
python src/main.py
```

When using the Poetry virtual environment, you can also start the server using the CLI command `python src/cli.py start [OPTIONS]`, where [OPTIONS] includes the API key id, API key, the Cortex PAPI server URL, and the log level.

Use the CLI

From the CLI, you can run three commands.

- **start**: Start the Cortex MCP server. Relevant only for the Poetry virtual environment.
- **update**: Any new or updated components provided by Cortex are automatically downloaded into the remote\_components folder. During each update, the folder is fully replaced and all existing contents are recreated. Do not add custom tools to this directory, as it is managed entirely by Cortex and is overwritten at every update.
- **version**: Display the current version of the Cortex MCP Server.

Additional information about the CLI is available in the README file located in the `src` directory.

### 3.4.1.2 | Configure the MCP client

Abstract

Configure your local MCP client to communicate with the Cortex MCP server.

After you have downloaded and installed the Cortex MCP server, you need to configure your local MCP client to communicate with the Cortex MCP server. The instructions below use Claude Desktop, but any MCP client can be used.

1. In the Claude Desktop app, navigate to Settings → Developer → Edit Config. The configuration file opens in your default text editor.

For reference, the file is located at:

- macOS: `~/Library/Application Support/Claude/clade_desktop_config.json`
- Windows: `%APPDATA%\Claude\clade_desktop_config.json`

2. Add the `mcpServers` configuration to the file. The examples below are provided for local client (Poetry virtual environment) and container (Docker). The exact details of your `mcpServers` configuration depend on your specific installation.

Poetry virtual environment

```
{
  "mcpServers": {
    "Cortex MCP Server": {
      "url": "http://127.0.0.1:8000"
    }
  }
}
```



```

    "command": "python",
    "args": [
        "/path/to/cortex-mcp/src/main.py"
    ],
    "env": {
        "CORTEX_MCP_PAPI_URL": "https://api.cortex.example.com",
        "CORTEX_MCP_PAPI_AUTH_HEADER": "<your_api_key>",
        "CORTEX_MCP_PAPI_AUTH_ID": "<your_api_key_id>",
        "MCP_TRANSPORT": "stdio/streamable-http"
    }
}
}
}
}

```

#### Docker Container

```
{
    "mcpServers": {
        "Cortex MCP Server": {
            "command": "docker",
            "args": [
                "run",
                "--env-file",
                "/path/to/.env",
                "-i",
                "--rm",
                "cortex-mcp"
            ]
        }
    }
}
```

3. Save the changes to the configuration file and restart Claude Desktop for the changes to take effect.

4. Verify the connection to the Cortex MCP server. You should see the Cortex MCP server running in the Developer settings and a hammer icon may appear in the input box, indicating the MCP tools are available.

#### 3.4.1.3 | Use the Cortex MCP server

##### Abstract

Use the MCP server to investigate and manage cases and issues from your local MCP client.

The Cortex MCP server provides built-in tools to manage cases and issues and conduct investigations.

Built-in tools include, but are not limited to:

- **get\_assets**: Fetch all assets, or a filtered subset of assets, based on criteria such as category, region or provider.
- **get\_assets\_by\_id**: Fetch detailed information about the asset specified by the asset ID.
- **get\_cases**: Fetch all cases, or a filtered subset of cases matching specific criteria such as domain, status, severity or specific case ID.
- **get\_issues**: Fetch all issues, or a filtered subset of issues matching specific criteria such as domain, severity, detection method or specific issue ID.
- **get\_assessment\_results**: Fetch the results of all or filtered compliance assessments from the Cortex platform.
- **get\_filtered\_endpoints**: Fetch a filtered list of endpoints managed by the XDR agents based on their status, XDR agent status, and other filters.

When you run the **update** command in the Cortex MCP server, new or updated tools provided by Cortex are automatically downloaded.

You also have the flexibility to create and customize your own tools to fit specific use cases and workflows. For more information, see Create custom Cortex MCP server tools.

##### Use case examples

##### **NOTE:**

The built-in tools retrieve information, but do not write to the tenant. You can create your own tools that include write actions. The examples below include both.

- Show me the top ten most critical cases and create a graphical representation for my manager to review.
- Give me the details for case ID 12345 and create a visual timeline.
- Isolate endpoint WIN-123 because it may be compromised.
- Retrieve full details for endpoint XXXX.
- Add a note to case 12345 saying “Escalated to Tier 2 for further investigation.”



#### 3.4.1.4 | Create custom Cortex MCP server tools

##### Abstract

Create your own customized tools to manage cases and issues.

You can build your own tools using OpenAPI or Python to manage cases, handle issues, and conduct investigations. More detailed information can be found in the README file located in the `src/usecase` directory. Tools are based on Cortex API endpoints.

##### NOTE:

Any new or updated components provided by Cortex are automatically downloaded into the `remote_components` folder. During each update, the folder is fully replaced and all existing contents are recreated. Do not add custom tools to this directory, as it is managed entirely by Cortex and is overwritten at every update.

##### OpenAPI

You can create an OpenAPI specification for a specific API endpoint.

1. Create a YAML file in the `/custom_components/openapi` directory with the name of the MCP component. For example:  
`custom_cortex_component.yaml`.
2. Base your custom OpenAPI component on the Cortex API documentation structure for a specific endpoint. We recommend viewing the built-in tools, located at `/builtin_components/openapi`, as a reference.
3. After you define the OpenAPI specification, the Cortex MCP server collects it automatically and it is ready for use.
4. Test your new MCP component by running the Cortex MCP server and writing a prompt that uses your new component.

##### Python

We recommend using Python for more complex MCP components that require custom logic. MCP components in Python are defined in a module.

1. Create a new Python file in the `/custom_components` directory.
2. Define a class that inherits from the `BaseModule` class with the required methods. We recommend viewing the built-in modules, located at `/builtin_components`, as a reference.
3. After you define a class, the Cortex MCP server collects it automatically and it is ready for use.
4. Test your new MCP component by adding an end-to-end test in the `tests/e2e` directory or run the MCP server and write a prompt that uses your new component.

## 3.5 | Users and Roles Management

##### Abstract

Configure and manage roles, users, and user groups, and set up authentication in Cortex AgentiX.

Learn how to configure and manage users, roles, and user groups. Assign roles and set up authentication for users.

#### 3.5.1 | Users and roles in Cortex AgentiX

##### Abstract

Set up and configure roles and user groups for the Cortex AgentiX tenant and Cortex Gateway. Configure authentication and manage users.

Cortex uses role-based access control (RBAC) to manage roles with specific permissions for controlling user access. RBAC helps manage access to components, so that users, based on their roles, are granted the minimal access required to accomplish their tasks.

##### Roles

Roles enable you to define permissions for specific components, such as cases and issues, playbooks, scripts, and jobs. For example, you can create a role that allows users to edit cases and issues, but not edit playbooks. You can create new roles or customize out-of-the-box roles.

Roles can also be used to define permissions for integration commands. On the Integration Permissions page, you can assign roles to specific integration instances (all commands for that instance) or specific integration instance commands. For example, you could assign the Generic Export Indicators Service integration instance the Account Admin role, or you could restrict certain commands in the Core Rest API to a specific role. For more information, see Integration Permissions.

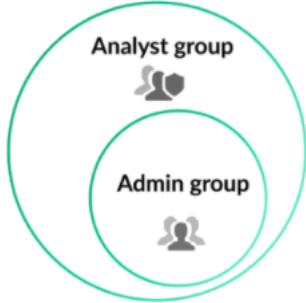
##### User groups

While roles can be assigned directly to users, we recommend creating user groups. Each user group has a single role associated with it, but each user group can contain multiple users, and user groups can be nested within each other, enabling you to further refine your RBAC requirements. Users can belong to several user groups.



#### Nested roles

Cortex AgentiX uses group nesting, where the group with higher permissions includes the permissions of the group with lower permissions, but as a subset of the group with lower permissions. For example, the Admin user group is included as a subset of the Analyst user group, as shown in the following graphic. The Admin role includes the permissions of the Analyst role.



For example, Content Developer and Analyst user groups include Employee user group permissions and are nested in the Employee user group.

#### Dataset access permissions

You can also set dataset access permissions using user roles or specific permissions using RBAC. Configuring administrative access depends on your organization's security requirements. Dataset permissions control dataset access for all components, while RBAC controls access to a specific component.

By default, when dataset access management is disabled, users have access to all datasets. If you enable dataset access management, you must configure access permissions for each dataset type and each user role. When a dataset component is enabled for a particular role, the Issues and Cases pages include information about datasets.

##### **NOTE:**

You can only set dataset access permission for a user role in the tenant. These settings are disabled in Cortex Gateway

For more information on how to set dataset access permissions, see [Manage roles in the Cortex AgentiX tenant](#).

#### Scope-Based Access Control (SBAC)

SBAC enables you to assign users to specific tags of different types in your organization. By default, all users have management access to all tags in the tenant. However, after an administrator assigns a management scope to a user (non-administrator), the user is then able to manage only the specific tags and their associated entities that are predefined within that scope. For more information, see

#### Authentication

You can create users in the Customer Support Portal or by using SAML Single Sign-On (SSO) in the tenant. After you create users, they authenticate by doing the following:

- Authenticate through the Customer Support Portal
- Authenticate by using SAML Single Sign-On (SSO) in the Cortex tenant

#### Manage users

By default, users do not have roles assigned and do not automatically have access to tenant data until you assign them a role or add them as members of a user group that has an assigned role.

### 3.5.2 | Roles management

#### Abstract

Configure roles in the Cortex tenant or Cortex Gateway.

You can assign the following permissions to various components in Cortex:

Permission	Description
None	No access to the specified component.
View	View, but cannot edit the specified component.



Permission	Description
View/Edit	View and edit the specified component.

#### Out-of-the-box roles

Cortex products include several out-of-the-box roles, such as:

Role	Type	Description
Account Admin	Predefined	<p>A super user role that is assigned directly to the user in Cortex Gateway or tenant and has full access to all Cortex products in your account, including all tenants added in the future. In Cortex Gateway, the Account Admin can assign roles for Cortex instances, and can also activate Cortex tenants specific to the product. This user has the same view/edit permissions in the tenant as the Instance Administrator.</p> <p><b>NOTE:</b></p> <p>The user who activated the Cortex product is assigned the Account Admin role.</p> <p>Only users with the Account Admin role can add or remove another Account Admin user role.</p> <p>You cannot edit this role. You can copy the role by saving it as a new role and then changing permissions.</p>
Instance Administrator	Predefined	<p>View/edit permissions for all components and access to all pages in the Cortex tenant. The Instance Administrator can also assign the Instance Administrator role to other users on the tenant. If the application has predefined or custom roles, the Instance Administrator can assign those roles to other users.</p> <p>You cannot edit this role. You can copy the role by saving it as a new role and then changing permissions.</p>
Viewer	Predefined	<p>Read permissions for all components and pages in the Cortex tenant.</p> <p>Cortex AgentiX products comes out-of-the-box with the following Viewer roles:</p> <ul style="list-style-type: none"> <li>•  Viewer role created in the tenant.</li> </ul> <p>This role is specific to the tenant. You can edit all permissions and delete the role (if not assigned to a user) in the tenant.</p>

#### NOTE:

By default, users do not have roles assigned. If no direct or user group role has been assigned, users don't have permission to view or edit data in the Cortex tenant.

#### Next steps

Before you start creating or customizing roles, do the following:

- Review the Role-based permissions topic.
- Decide whether you want to assign roles to users directly or through membership in user groups (recommended).

#### 3.5.2.1 | Role-based permissions in Cortex AgentiX

##### Abstract

Describes the role-based permissions available in Cortex AgentiX

When creating or editing a role, you can set permission levels (RBAC) for specific components (such as playbooks, scripts, jobs, etc.), set page access, and set up Dataset permissions.

In the Cortex AgentiX tenant, you can set permission levels for each role by going to Settings → Configurations → Access Management → Roles and then editing or creating a new role.

If using Cortex Gateway, go to Permission Management → Roles.

#### NOTE:



You can only create, edit, copy, or delete a role if you have administrator (Instance/Account Admin) permissions. You cannot change the permissions of predefined roles, such as Instance Administrator.

Each role contains the following tabs:

#### The Components tab

The Components tab includes the following areas where you can define permissions.

#### Dashboards & Reports

Component	Description
Dashboards	<p>Limits permissions for managing dashboards, such as creating and editing dashboards in the Dashboard Manager.</p> <p><b>NOTE:</b></p> <p>Users can't edit a predefined dashboard, but they can create a new one based on a dashboard template. Custom dashboards can only be viewed and edited by the dashboard creator.</p>
Ingestion Monitoring	<p>The Data Ingestion dashboard provides an overview of the ingestion status of all sources and a view of the daily quota consumption.</p> <p>Limit permissions for the Data Ingestion dashboard.</p> <p><b>NOTE:</b></p> <p>Users can't edit the Data Ingestion dashboard.</p>
Reports	Limits permissions for managing reports, such as creating and editing reports on the Reports Template page.

#### Cases & Issues

Component	Description
Cases and Issues	<p>Limits permissions for taking any action on cases and issues, such as creating cases, responding to cases and issues, and adding a trigger playbook. For more information about Cases and Issues, see Investigate cases.</p> <p><b>NOTE:</b></p> <p>If SBAC is set to Restrictive mode, users who don't have all the tags shouldn't be able to read or edit the parent case (fields or context). For more information on setting restrictive mode, see Configure server settings.</p> <p>However, if users are assigned all the tags on the child issue and have View/Edit permissions on Cases and Issues, users can trigger a playbook that could potentially change the parent case (even though users should not be able to do so according to SBAC).</p> <p>In this case, you can grant Add Trigger Playbook permissions, so users can bypass SBAC on the parent case fields and context data. For more information about updating fields in a playbook, see Update issue fields.</p>

#### Investigation & Response



Section	Component	Description
Search	Query Center	<p>Limit permissions for the Query builder, Query Center, and scheduled queries, such as Correlation Rules.</p> <p>The Query Center displays information about all queries that were run in the Query Builder. From the Query Center, you can manage your Cortex Query Language (XQL) and Graph Search queries by viewing query results, running queries, adjusting queries, and scheduling when a query runs. For more information about XQL, see <a href="#">Get started with XQL</a>.</p> <p><b>NOTE:</b></p> <p>Editing Correlation Rules requires View/Edit permissions for both the Query Center and Rules in the Threat Management section (under Detections (see below)).</p>
	Personal Query Library	<p>Cortex AgentiX provides a personal query library for saving and managing your XQL queries. When creating a query in XQL Search or managing your queries from the Query Center, you can save queries to your personal query library and decide whether to share them.</p> <p>For more information about the personal query library, see <a href="#">Manage your personal query library</a>.</p>
Automations	Playbooks	<p>Limits permissions for creating, editing, and deleting playbooks.</p> <p><b>NOTE:</b></p> <p>You can also add, change, and remove roles from a playbook by clicking the settings wheel on the Playbook Starts task and adding a role in the General tab.</p> <p>User role permissions are by default set up to view playbooks. You can't set up a role without any access to scripts.</p> <p>For more information about playbooks, see <a href="#">Playbooks overview</a>.</p>
	Scripts	<p>Limits permissions for managing scripts. If the role has read/write permissions, you can enable user roles to create scripts that run as a Super User.</p> <p><b>NOTE:</b></p> <p>You can't edit out-of-the-box scripts, but you can create a new one based on the script.</p> <p>User role permissions are by default set up to view scripts. You can't set up a role without any access to scripts.</p> <p>For more information about scripts, see <a href="#">Use existing scripts</a>.</p>



Section	Component	Description
	Playground	<p>Limits the ability to run commands, APIs, and scripts on the Playground page.</p> <p>The Playground is a non-production environment where you can safely develop and test data, such as scripts, APIs, and commands. It is an investigation area that is not connected to a live (active) investigation.</p> <p>You can't select a view permission for the Playground.</p> <p>For more information about the Playground, see <a href="#">Use the War Room in an investigation</a>.</p>

#### Threat Management

Section	Component	Description
Detections	Rules	<p>Limit the ability to manage Correlation Rules, such as create, edit, export, and delete them.</p> <p><b>NOTE:</b></p> <p>Editing Correlation Rules requires View/Edit permissions for Query Center (see above) and Rules.</p> <p>For more information about Correlation Rules, see <a href="#">Manage correlation rules</a>.</p>
Threat Intelligence	Threat Intelligence	Limits all actions under the Threat Intel Management section.

#### Marketplace

Component	Description
Marketplace	<p>You can set the following permissions for Marketplace.</p> <ul style="list-style-type: none"> <li>None: The user role is not able to view Marketplace.</li> <li>View: The user role can view, but not take any action in Marketplace.</li> <li>View/Edit: The user role can install, upgrade, downgrade, and delete content packs in Marketplace.</li> </ul> <p>For more information about Marketplace, see <a href="#">Marketplace FAQs</a>.</p>

#### Configurations

Section	Component	Description
General Setting	Auditing	<p>Whether a user role can access the Management Audit Logs page.</p> <p>For more information, see <a href="#">Forward logs and data from Cortex AgentX to external services</a></p>
General Setting	Alert Notifications	Whether a user role can limit user log notifications. For more information, see <a href="#">Data and log notification formats</a> .
	General Configuration	Limits the ability to change server settings, such as timezone, timezone format, and email contacts. For more information, see <a href="#">Configure server settings</a> .



Section	Component	Description
Access Management	Access Management	Limits the ability to manage users, roles, scopes, user groups, and authentication. For more information, see Users and roles in Cortex AgentiX.
Data Collection	Data Sources	Limits the ability to manage data sources, such as add, edit, and delete. For more information about data sources, see Add a new data source or instance.
	Integrations	Limits the ability to add an engine
Data Management	Data Management	<p>Limits the ability to use the following features:</p> <ul style="list-style-type: none"> <li>• Dataset Management</li> <li>• Parsing Rules</li> <li>• Data Modelling Rules</li> <li>• Event Forwarding</li> </ul> <p>For more information, see Data management.</p>
Integrations	Public API	<p>Whether a user role can access the API Keys page. View/Edit enables the user role to manage API keys, including creating, editing, and deleting.</p> <p><b>NOTE:</b></p> <p>If you select None, the user role can still use the API, but they cannot view API keys in the UI.</p> <p>For more information about how to create APIs in Cortex AgentiX, see Cortex AgentiX API.</p>
	Threat Intelligence	Limits the ability to create, edit, and delete Threat Intelligence Integrations on the Settings à Configurations à Integrations à Threat Intelligence page. For more information, see External integrations.
	Long Running HTTP Integrations Configuration	Limits the ability to create, edit, and delete long-running integrations on the Settings à Configurations à Integrations à External Dynamic List Integration page. For more information, see Manage external dynamic lists.
	Integrations Permissions	<p>Enables you to set permissions on the Integration Permissions page. Integration permissions enable you to assign different permission levels for the same command in each instance.</p> <ul style="list-style-type: none"> <li>• None: The user role cannot view the page.</li> <li>• View: The user can view the page.</li> <li>• View/Edit: The user can view and edit permissions.</li> </ul> <p>For more information about integration permissions, see Configure integration permissions.</p>
	Credentials	Whether a user role can add, edit, or delete integration credentials. For more information, see Manage credentials
Object Setup	Case Properties	<p>Limits permissions to edit Settings à Configurations à Object Setup à Cases à Properties.</p> <p>Users can create custom cases statuses and custom resolution reasons that are tailored to their workflow. For more information, see Create custom case statuses and resolution reasons.</p>



Section	Component	Description
	Exclusion list	<p>Indicators added to an exclusion list are disregarded by the system and are not created or involved in automated flows such as indicator extraction.</p> <p>Limits permissions when editing, creating, or deleting an indicator in an exclusion list.</p>
	Fields and Types	Whether a user can add, edit, or delete fields and types for indicators, cases, and issues. For more information, see <a href="#">Customize case fields and layouts</a> .
	Layouts	Whether a user can add, edit, or delete layouts for indicators, cases, and issues. For more information, see <a href="#">Customize case fields and layouts</a> .

Help

Component	Description
Support	Whether the user role can send a support case. For more information, see <a href="#">In-product support case creation</a> .

Datasets tab

Under the Datasets (Disabled) tab, you have the following options for setting the Cortex Query Language (XQL) dataset access permissions for the user role:

- Set the user role with access to all XQL datasets by leaving the dataset access management as disabled (default).
- Set the user role with limited access to certain XQL datasets by selecting the Enable dataset access management toggle and selecting the datasets under the different dataset category headings.

For more information about Datasets, see [Dataset management](#).

### 3.5.2.2 | Manage roles in the Cortex AgentiX tenant

Abstract

Manage roles in Cortex AgentiXCortex AgentiX tenant.

On the Roles page, you can view all roles in Cortex AgentiX, whether they are custom roles, who created the role, when it was created, and additional information about the roles. When right-clicking on a role, you can edit the role and permissions.

Cortex AgentiX includes the following role types:

- **Predefined roles:** Includes Account Admin and Instance Administrator roles. Permissions cannot be changed. You can create a duplicate of these roles, but you cannot remove them.
- **Custom roles:** Includes out-of-the-box roles and custom roles.

When right-clicking a role, you can perform several actions, such as editing a role, saving it as a new role, and removing a role (deleting a role not assigned to a user).

If you want to edit a role, right-click the relevant user role and select Edit Role. If you want to create a new role based on an existing role, right-click the relevant user role and select Save As New Role.

Create a role

The roles you create provide more granular access control. You can add as many new roles as you need and combine them with user groups. When you create or edit a role, you can perform activities such as adding permissions and permission levels.

To create, edit, or delete a role, you must have administrator permissions.

**TIP:**

For analysts, we recommend limiting specific permissions, such as:



- Removing the ability to install, delete, Marketplace/Data Sources, which should be reserved for engineers and administrators. We recommend setting these permissions for analysts to None or View.
- Removing access to API keys. Under CONFIGURATIONS, set the Public API access to None or View. If you select None, the user role can still use the API, but they cannot view API keys in the UI.

1. In the Cortex AgentX tenant, select Settings → Configurations → Access Management → Roles → New Role.

**TIP:**

We recommend making a copy of out-of-the-box roles and editing the copies, rather than creating new roles, to avoid missing any important permissions.

2. Add the Role name and a meaningful Description.

3. In the Components tab, add the permissions as required. For more information, see Role-based permissions in Cortex AgentX.

4. Under Datasets (Disabled), you have two options for setting the Cortex Query Language (XQL) dataset access permissions for the user role:

- Set the user role with access to all XQL datasets by leaving the dataset access management as disabled (default).
- Set the user role with limited access to certain XQL datasets by selecting the Enable dataset access management toggle and selecting the datasets under the different dataset category headings.

5. Save the role.

6. You can create user groups and add roles to them (recommended), assign roles directly to users after they have been added, or both.

### 3.5.3 | User group management

#### Abstract

Create user groups and assign roles and users to further refine your requirements.

Users are assigned roles and permissions either by being assigned a role directly or by being assigned membership in one or more user groups. A user group can only be assigned to a single role, but users can be added to multiple groups if they require multiple roles. You can also nest groups to achieve the same effect. Users who have multiple roles through either method will receive the highest level of access based on the combination of their roles. The same principle for users with multiple roles is followed for both the Role-Based Access Control (RBAC) access permissions and the Scope-Based Access Control (SBAC) granular scoping, so that users receive the highest level of access by combining their roles.

#### Example 9.

- Joe has an Analyst role and is a member of the Tier-1 Analyst user group, which is assigned the Triage role. Joe has the permissions of the Analyst role and the Triage role. Joe is assigned 2 roles, and has the highest permission based on the combination of both roles.
- John is a member of two user groups - Tier-1 Analyst and Tier-2 Analyst. One group is configured to use the Triage role and the other group is configured to use the Incident Response role. John is assigned both roles and has the highest permissions based on the combination of all roles.
- Jack is a member of the Tier-2 user group, which has an Incident response role. This user group is included in a Tier-3 user group (Threat Hunter role), added as a nested group. Jack is assigned both roles and has the highest permissions based on the combination of all roles.

On the User Groups page, you can create a new user group for several different system users or groups. You can see information including the details of all user groups, the roles, nested groups, IdP groups (SAML), and when the group was created/updated.

You can also right-click in the table to edit, save as a new group, remove (delete) a group, and copy text to the clipboard.

**NOTE:**

You can create user groups in the tenant or Cortex Gateway. User groups created in Cortex Gateway cannot be mapped to SAML groups. Only user groups created in the tenant support SAML group mapping and scoring. We recommend creating user groups in the Cortex tenant because:

- User groups are available for all tenants, and you may want different user groups in different tenants, such as dev/prod.
- You can apply granular scoping for a user role by granting access only to the relevant data that the user requires for their designated role in the tenant. You also need to enable scope-based access control in the Server Settings page. For more information, see Manage user scope.

Before configuring SBAC, ensure that you review Understand scoping in the Manage user scope section.

#### How to create a user group

1. Go to Settings → Configurations → Access Management → User Groups.

If creating in Cortex Gateway, go to Permission Management → User Groups.

2. To create a new user group for several different system users or groups, click New Group, and add the following parameters:



Parameter	Description
Name	Name of the user group.
Description	Description of the user group.
Group for product	(Cortex Gateway only) If you have multiple products, select the relevant Cortex product.
Role	Select the group role associated with this user group. You can only have a single role designated per group.  In Cortex Gateway, you can only select either Instance Administrator or a custom role created in the Gateway.
Users	Select the users you want to belong to this user group.  <b>NOTE:</b> If users have been created in the CSP, but you want them to access the tenant through SSO only, skip this field and add only SAML group mapping after SSO is set up, otherwise, users can access the tenant through both the CSP and SSO. If you have not yet created any users, skip this field and add them later. See Set up authentication .
Nested Groups	Lists any nested groups associated with this user group. If you have an existing group, you can add a nested group.  User groups can include multiple users and nested groups, which inherit the permissions of parent user groups. The user group will have the highest level of permission.  For example: <ul style="list-style-type: none"> <li>• Group A has Tier-1 Analyst permissions</li> <li>• Group B has Tier-2 Analyst permissions</li> </ul> If you add Group A as a nested group in Group B, Group A inherits Group B's permissions (Tier-1 and Tier-2 permissions).  In Cortex Gateway, you can only add user groups that are created in Cortex Gateway.
SAML Group Mapping	(Relevant when creating a user group in the Cortex tenant only.)  Maps the SAML group membership to this user group. For example, you have defined a <code>Cortex Admins</code> group. You need to name this group exactly how it appears in Okta.  You can add multiple groups by separating them with a comma.  <b>NOTE:</b> When using Azure AD for SSO, the SAML group mapping needs to be provided using the group object ID (GUID) and not the group name.  If you have not set up SSO in your tenant, skip this field and add it later. After you have added it, follow the procedure relevant to your IdP. For example, see Set up Okta as the identity using SAML 2.0 .

3. (Optional) When creating the user group in the tenant, configure granular scoping for the user group.

If creating the user group in the Cortex Gateway, you can skip this step, as scoping is only supported in the tenant.

a. Click the Scope tab.

b. Expand the scoping areas that you want to grant the user role access to in the tenant by clicking the chevron icon (>) beside the scoping area title, and make any changes required. The following table explains the options available to configure:



Scoping Area    Granular Scoping Configurations	
Assets	<p>Set the Scope by selecting one of the following:</p> <ul style="list-style-type: none"> <li>• No assets: No asset is accessible.</li> <li>• All assets: Defines access to all assets.</li> <li>• Select asset groups: Defines access to the specific assets associated with the Asset Groups selected, and to view all their related cases, issues, and findings for these specific assets and Asset Groups. Under Select asset groups, define the specific asset groups that you want to grant access. Only Asset Groups relevant for scoping are listed, which are asset groups that are using only the asset attributes listed in Manage user scope (under Understand scoping â Scoping Areas â Assets).</li> </ul> <p>The scoping of assets also affects the scoping of cases, issues, and findings.</p> <p><b>NOTE:</b></p> <p>Visibility of Security domain Issues that refer to assets with agents is controlled by the Endpoints scoping configuration.</p>
Cases and Issues	<p>Set the Scope by selecting one of the following:</p> <ul style="list-style-type: none"> <li>• No cases and issues: Defines access to no cases and issues.</li> <li>• All cases and issues: Defines access to all cases and issues. Users can view cases or issues referencing assets within their scope. Use the Assets section to define which assets are in scope.</li> <li>• Select domains: Defines access to the domains selected to view their related cases and issues. Under Select domains, define the specific domains that you want to grant access.</li> </ul> <p>Users can only view cases or issues referencing assets and endpoints within their scope. Use the Assets section to define which assets are in scope.</p> <p>When selecting All cases and issues or Select domains, you can separately configure access to issues and cases that lack an asset reference or where the referenced asset is not in All Assets and All Endpoints inventories. To provide access, select the Allow access to cases and issues that are not referencing known assets or endpoints checkbox.</p>
Endpoints	<p>Set the Scope by selecting one of the following:</p> <ul style="list-style-type: none"> <li>• No endpoints: Defines access to no endpoints with no ability to view their related agent management and enterprise policies.</li> <li>• All endpoints: Defines access to all endpoints with the ability to view their related agent management and enterprise policies. This configuration can impact the visibility of related Security domain Cases and Issues, but will not affect asset visibility.</li> <li>• Select specific (at least one required): Defines specific access to all endpoint groups by selecting Endpoint Groups or all endpoint tags by selecting Endpoint Tags to view their related agent management and enterprise policies. This configuration can impact the visibility of related Security domain Cases and Issues, but will not affect asset visibility.</li> </ul>

#### IMPORTANT:

By default, Enable Scope Based Access Control is disabled in Settings â Configurations â General â Server Settings, and granular scoping is not enforced. Before enabling SBAC, we recommend that an administrator or a user with Access Management permissions first ensures that the users, user groups, and API Keys defined in Cortex AgentIX are granted the required access by assigning the relevant scopes. For more information, see [Manage user scope](#).

4. Click Create to create the user group.

#### 3.5.4 | User management

##### Abstract

Manage users in Cortex Gateway or the Cortex tenant.

You can manage users in Cortex Gateway or the Cortex tenant. At the Users tab in the Permissions page (Cortex Gateway â Permission Management+Permissions) or the Users page (Settings â Configurations â Access Management â Users) in the tenant, you can see user information, including:



Name	Description
User Type	Indicates whether the user was defined in Cortex using the CSP, SSO using your organization's IdP, or both CSP/SSO.
Direct Role	<p>Displays the name of the role assigned specifically to the user, not inherited from somewhere else, such as a user group.</p> <p>In Cortex Gateway, select the arrow next to the name of the user to see the user's roles and the tenants the user has access to.</p> <p>In the Cortex tenant, the direct role is the role assigned to the user in the tenant.</p> <p>When the user has no access permissions assigned specifically to them, the field displays No-Role.</p> <p><b>NOTE:</b></p> <p>If a user does not have a direct role or user group assigned, the user is revoked and is not saved in the Cortex Gateway.</p>
Groups	<p>Lists the user groups to which the user belongs.</p> <p>If a user is assigned to multiple user groups, which are mapped to different roles, or if the user is assigned to nested user groups, the user inherits the permissions of parent user groups and has the highest level of privileges based on the combination of roles.</p> <p>Any group imported from Active Directory has the letters AD added beside the group name.</p> <p><b>NOTE:</b></p> <p>If a user does not have a direct role or user group assigned, the user is revoked and is not saved in the Cortex Gateway.</p>
Group Roles	<p>Lists the different group roles based on the groups the user belongs to. When you hover over the group role, the group associated with this role is displayed.</p>
Scope	<p>Only visible if Scope-Based Access Control is enabled for the tenant.</p> <p>Lists the scope assigned to the user either directly or through a group, based on tags. The family includes the tag types and the related tags of the selected family.</p>
Status	<p>Displays whether the user is Active or Inactive.</p>
Phone number	<p>Relevant only in the Cortex tenant.</p> <p>Displays the user's phone number. Including the user's phone number enables playbooks and scripts to trigger direct analyst communication by phone.</p>

### 3.5.4.1 | Manage users in the Cortex AgentiX tenant

#### Abstract

On the Users page, view user information and edit users and their roles.

To access Cortex AgentiX users must be created in the Customer Support Portal (CSP) and added to the tenant or created via SSO. When logging into Cortex AgentiX users must have a direct role or a user group role. If no role is assigned either directly or via a user group, they do not have access to the tenant.

**NOTE:**

To remove users that were added to your CSP account, you need to do this in the CSP and not in the tenant or Cortex Gateway.

When right-clicking a user on the Users page, you can do the following:



- Add/update the user role

- Edit user permissions

View the user's details. You can add a phone number, which enables playbooks and scripts to trigger direct analyst communication by phone.

- Remove User Role
- Hide the User
- Deactivate User
- Import user roles

#### Add/update user roles

You can update user roles for one or multiple users. You can add/update the following user roles:

- **Pre-Defined roles:** Instance Administrator and Account Admin. If you want to remove the Account Admin role from a user, you need to remove it in Cortex Gateway.
- **Custom roles:** Includes out-of-the-box roles and roles created in Cortex Gateway or the tenant.

#### **NOTE:**

To update the permissions attributable to each role, change them in the Roles tab or Cortex Gateway.

If users have been created in the CSP, but you want them to access the tenant through SSO only, you should not assign a direct role. If you sign a direct role, users can access the tenant through both the CSP and SSO.

1. Go to Settings → Configurations → Access Management → Users, and do one of the following:

- To edit one user, right-click the user's name and select Edit User Permissions.
- To edit multiple users, select multiple users, right-click, and select Edit Users Permissions.

2. In the Role field, select one of the pre-defined or custom roles.

- Pre-Defined Roles
- Custom roles

If no role is assigned either directly or via a user group, users do not have view or edit permissions in Cortex AgentIX.

The Show Accumulated Permissions field shows the roles and user groups assigned to the user. You can also select the specific roles assigned to the user, which enables you to compare available permissions based on the roles selected. This can help you understand how the role permissions for a particular user are built. For example, if you need to isolate a specific component, the permissions are provided by a particular role or user group.

3. Add User Groups if required.

4. (Optional) If Scope-Based Access Control is enabled for the tenant, click Scope and select a tag family and the corresponding tags.

Guidelines for selecting tags

Keep in mind the following:

- Roles defined as administrator or a part of the admin group can't be scoped.
- If you select a tag family without specific tags, permissions apply to all tags in the family.
- The scope is based only on the selected tag families. If you scope only based on tags from Family A, then Family B is disregarded in scope calculations and is considered as allowed.

5. You can also set user access permissions for the various Cortex Query Language (XQL) datasets.

6. Save the user role.

#### Import multiple user roles

Rather than assigning roles to each user, you can import multiple user roles to add users who have a Customer Support Portal account and assign them existing predefined or custom roles in Cortex AgentIX. On the Users page, when clicking Import Multiple User Roles, download the example file and replace the file contents with the data to upload. The following values must be included:

Parameter	Value
User email	The email address of the user belonging to the Customer Support Portal account that you want to import.



Parameter	Value
Role name	The name of the role that you want to assign to this user. The role must already be created in Cortex AgentiX.
Is an account role (default=false)	Determines whether the user role is created in Cortex Gateway or the tenant. If defined in Cortex Gateway, set the value to True; otherwise, the value is set to False (default).

#### Remove a user role

If a user has a role in the tenant (not Account Admin), you can remove their user permission to access the tenant. If no direct or user group role has been assigned, the user role displays No Role, and has no permission to view or edit on Cortex AgentiX.

1. In the Users tab, right-click the user's name and select Remove User Role.
2. Confirm that you want to Remove the user role.

#### Deactivate users

Users should be deactivated to temporarily remove user access to the Cortex AgentiX tenant. All user information is maintained for deactivated users. Users should be permanently removed from the CSP if they no longer need access to Cortex products. If you want to remove a user from the CSP, you need to reassign issues and tasks to another user before removing them.

#### NOTE:

You cannot deactivate a user who has an Account Admin role. If you want to deactivate users from all tenants, deactivate them in Cortex Gateway.

The user will be deactivated in the tenant, but may still be active in other tenants. If you want to deactivate the tenant for multiple tenants, deactivate the user in Cortex Gateway.

Go to the Cases page and search for Status != Resolved AND Assignee = <name of assignee> to find any cases the user is assigned and reassign.

If the user is assigned to cases and issues, these assignments do not automatically change when the user is removed or deactivated. We recommend changing issues and task assignments manually before removing or deactivating users.

Any reports the user has created remain available. Reports are not owned by specific users and can be edited or deleted by other users.

#### NOTE:

When you remove a role, the role associated with the API keys is deleted.

- If more than one role was associated with the API key, a yellow warning symbol appears next to the API key in the API key table. When you hover over the symbol, a message indicates that some of the roles associated with the API key have been deleted.
- If all roles associated with the API key are removed, a red warning symbol appears next to the API key in the API key table. When you hover over that symbol, a message indicates that the key is no longer usable because it does not have a role associated with it. The API key is still visible in the API table, but it cannot be assigned.

When a user is deactivated, API keys that the user created are not revoked.

Before you deactivate a user, reassign open cases and issues to another user by going to the Cases page and search for Status != Resolved AND Assignee = <name of assignee> and reassign.

#### How to deactivate users

1. From the Users page, right-click the user's name and select Deactivate User.
2. In the dialog box, Deactivate the user.

#### Hide users

Hides users from the user list in the tenant. This is useful when you have users who are not related to your Cortex tenant and will not be designated with a role, such as CSP Super Users, and you want to hide them from the list.

You cannot view the user or search for the user when hidden. To hide a user, select the name, right-click the user's name, and select Hide user. The user is no longer displayed when the table is configured to hide hidden users (default). To view the user, select Actions → Show Hidden users. If you want to remove the hidden designation, right-click the user's name and select Unhide user.

#### 3.5.4.2 | Manage user scope

##### Abstract

Learn about Scope-Based Access Control (SBAC) and how to assign users to specific scoping areas in your organization.



## PREREQUISITE:

- Configuring user scopes in Cortex AgentiX Access Management requires View/Edit RBAC permissions for Access Management (under Configurations). Account Admin and Instance Administrator roles are granted this permission by default. For more information, see *Predefined user roles* in Set up users and roles.
- By default, Enable Scope Based Access Control is disabled in Settings → Configurations → General → Server Settings, and granular scoping is not enforced. Before enabling SBAC, we recommend that you first ensure that the users, user groups, and API Keys defined in Cortex AgentiX are granted the required access by assigning the relevant scopes.

Review the following topics:

- Set up users and roles
- User group management
- Assign user roles and groups
- Manage user roles and access management

### What is SBAC?

Cortex AgentiX enables you to use Scope-Based Access Control (SBAC) in combination with Role-Based Access Control (RBAC) to define precise access controls according to your organization's security policies. While RBAC defines what a role can access and the actions that can be performed, SBAC determines the specific data and content displayed when accessing these areas and performing those actions.

Users with Access Management permission apply scopes to limit the data and content that users can be granted access to in Cortex AgentiX, which are divided into different scoping areas. The scoping areas include Assets, Cases and Issues, and Endpoints, which can be applied as relevant to the enforcement area or entity. For example, an Investigator role might have access to asset information based on the RBAC permissions, but the SBAC granular scoping configuration could limit that investigator's view and control to only assets within a particular scoping area. This hybrid approach ensures scalability and granular control, significantly strengthening system security by ensuring only authorized users are granted access to the relevant data that the user requires for their designated role.

Granular scoping for all scoping areas is configured in users, user groups, or API Keys according to the designated user role. Users are granted granular scoping access based on the user role assigned to them either in a user group or directly.

### Things to consider before configuring SBAC

Before you begin setting Scope-Based Access Control (SBAC) granular scoping, consider the following information:

- SBAC is disabled by default, which means that users have access to all content and data in the areas they have access to according to the RBAC permissions defined in their role.
- To best address Cases that span across all scopes, we recommend that there always be designated users with full access to all cases, issues, assets, and findings.
- Policies and playbook execution can affect items outside the user's scope, even though scoped users can't view them. As a result, we recommend that users who write policies be granted access to all relevant policy assets, so they can review the effects of the policies.
- Some areas and features in Cortex AgentiX do not comply with SBAC. In these cases, use RBAC permissions to restrict access. For more information, see Functional areas that respect and don't respect SBAC.
- Respecting SBAC has some performance overhead when opening the Cases, Issues, Findings, and Assets tables, which can take more time.
- In Reports, SBAC applies when a report is manually generated, not when it is accessed in any other way. Scheduled reports do not run in any user context and are not subject to SBAC.
- For users who upgraded from a previous version of Cortex AgentiX to the current version, see the What's New in Cortex XSIAM 3.x Guide for specific changes that you should know about.

### Understand scoping

#### Scoping areas

User Groups, Users, and API Keys can be scoped according to the following scoping areas:



- **Assets:** Provides access to the assets associated with asset groups, and enables you to access their related cases, issues, and findings. When using asset groups, you can limit access based only on this list of attributes: Asset Class, Category, Provider, Region, Organization, Realm, Business Application Names, Kubernetes Cluster, Kubernetes Namespace, Code Repository, and Asset Tags.
  - When you create or edit an Asset Group, the changes are applied immediately to new assets and to existing assets that have been updated. Yet, it can take a few hours for the changes to appear on existing assets that have not been updated.
- **Cases and Issues:** Provides access to domains to view their related cases and issues.
- **Endpoints:** Applies scoping on an endpoint as an entity and provides access to Endpoint Groups and Endpoint Tags to view their related agent management and enterprise policies.

**NOTE:**

This configuration can impact the visibility of the related Security domain in the Cases and Issues scope area, but will not affect asset visibility.

Scoping Behaviors

- When applicable, all conditions must be met to apply the scope configuration. For example, an issue with an affected asset is accessible only if the asset is in scope and the issue's domain is in scope.
- SBAC allows viewing cases and issues with no affected assets or endpoints, or when at least one affected asset is in the user's scope. The user can see all affected assets, including those not in scope, but won't be able to see more details about the assets not in scope, including opening their card.
- Cases and Issues of deleted assets do not have affected assets and so are not affected by asset-led SBAC or Endpoints, and are only based on the Cases and Issues domain.
- SBAC allows viewing cases where at least one of its issue domains is in the user's scope. The user can see all issues, including those not in scope, but won't be able to see more details about the issues not in scope, including opening their card.
- The behavior of cases and issues with affected endpoints depends on the Endpoint Scoping mode.
- XQL queries that use the `cases`, `issues`, `findings`, and `asset_inventory` datasets respect only the Assets scoping area configurations.

Functional areas that respect and don't respect SBAC

It is important to review both the functional areas and features in Cortex AgentiX that are respected and not fully respected so you can decide what actions to take in your tenant.

Functional areas respected

Scope-Based Access Control (SBAC) applies to the following functional areas in Cortex AgentiX:

**IMPORTANT:**

Some areas and features in Cortex AgentiX do not respect SBAC. In these cases, use RBAC permissions to restrict access.

Functional Area	Description	Related Scoping Area
Cases, Issues, Findings, and Assets tables	View and manage cases, issues, findings, and assets, and take actions in these tables.	<ul style="list-style-type: none"> <li>• Assets</li> <li>• Cases and Issues</li> </ul>
Dashboard and Reports	Scoping takes place only on the following: <ul style="list-style-type: none"> <li>• XQL-related widgets based on XQL queries that use the <code>cases</code>, <code>issues</code>, <code>findings</code>, and <code>asset_inventory</code> datasets, and respect only the Assets scoping area configurations.</li> </ul> <p><b>NOTE:</b></p> <p>XQL-based dashboard widgets may require a few hours to initially reflect changes to the list or definitions of asset groups used for scoping. To view the most current data immediately, refresh the dashboard or its XQL widgets.</p>	<ul style="list-style-type: none"> <li>• Assets</li> <li>• Cases and Issues</li> </ul>
Public APIs	Public APIs that access Cases, Issues, Findings, and Assets information respect Scope-Based Access Control (SBAC).	<ul style="list-style-type: none"> <li>• Assets</li> <li>• Cases and Issues</li> </ul>



Functional Area	Description	Related Scoping Area
Cortex Query Language (XQL)	<p>When using XQL with <code>cases</code>, <code>issues</code>, <code>findings</code>, and <code>asset_inventory</code> datasets, keep in the mind the following:</p> <ul style="list-style-type: none"> <li>XQL respects asset-led SBAC when accessing these datasets, including when using XQL queries and XQL widgets.</li> <li>XQL queries that use these datasets, respect only the Assets scoping area configurations.</li> </ul> <p><b>NOTE:</b></p> <p>For Cases and Issues domains, a workaround is to create a Dataset View for each required combination of domains, and allow the relevant entity access only to this Dataset View, not to the underlying <code>cases</code> and <code>issues</code> datasets.</p>	Assets

#### SBAC not fully respected functional areas

Ensure that you review the points below that explain the main functional areas with limitations with respecting SBAC, so you can decide how to handle this in your tenant. A suggested action is provided when applicable.

- Access to datasets: Access to the `alerts` and `incidents` datasets do not support SBAC. As a result, consider limiting users from accessing these datasets by excluding access to the datasets mentioned above using Dataset Views, and only enabling access to `cases` and `issues` datasets that respect SBAC.
  - Graph Search: Graph Search does not support SBAC. It is currently a Beta feature and is only available in the tenant using a feature flag.
  - Command Centers: Aggregate numbers in Command Centers can also sum up data that is not in the user scope. When pivoting from Command Centers to the Cases, Issues, Findings, and Assets tables, these tables do respect SBAC. We recommend limiting the users who access Command Centers, and these users should be granted a broader scope. For all other users, disable access in RBAC settings (Dashboards & Reports â> Command Center Dashboards).
  - Timeline widget
- As a workaround, you can disable access through RBAC permissions by disabling Dashboards (Dashboards & Reports â> Dashboards).
- Notification Center
  - Agent Installation widget: This widget is not available for scoped users.
  - Drop-downs of cases and issues domains: Drop-downs of these domains display all domains.
  - KSPM dashboard: Users can access all information on the dashboard when their user access is scoped to view All assets or assigned to the Instance Administrator role. Otherwise, users with granular scoping set to No assets or Select asset groups will have limited access to the dashboard. For more information on the KSPM dashboard, see Predefined dashboards.

[Feature Change] Visibility for cases and issues without Inventory Assets

#### IMPORTANT:

Action Required: Recent security enhancements enforce stricter default permissions for cases and issues that lack specific asset or endpoint references. This notice explains how to include access to these items if your users' visibility has been impacted.

To improve data security, Cortex AgentIX now restricts access by default for cases and issues that do not reference a specific asset or that involve assets not found in your standard inventories.

If your users previously relied on broad access to these items, an administrator or a user with access management permissions must manually enable the new setting to to include access to these cases and issues:

- Choose one of the following:
  - To edit the role for a user or user group, select Settings â> Configurations â> Access Management.
  - To edit the role of an API key, select Settings â> Configurations â> Integrations â> API Keys.
- Edit the relevant User, User Group, or API Key.
- In the Scope tab, under Cases and Issues, enable the checkbox: Allow access to cases and issues that are not referencing known assets or endpoints.
- Save your changes.

Once enabled, an (Extended) label appears next to the scope level.



Granular scoping is configured in users, user groups, or API keys, and applied to the user roles assigned. Users are then granted granular scoping access according to the user roles assigned to them in a user group or directly. The instructions below explain how to configure granular scoping according to Palo Alto Networks best practices.

Granular scoping is disabled and not enforced in Cortex AgentIX by default. Before enabling SBAC, we recommend that an administrator or a user with Access Management permissions first ensure that the users, user groups, and API Keys defined in Cortex AgentIX are granted the required access by assigning the relevant scopes. This user can then assign a scoping area to a Cortex AgentIX user (non-administrator), so the non-administrator user can manage only the specific scoping areas that are predefined within that scope.

Any changes made to the granular scoping of a user, user group, or API key are recorded on the Management Audit Logs page (Settings → Management Audit Logs). These events are categorized with the Type set to Permissions and the Subtype set to Scope Edit.

**NOTE:**

Make sure to assign the required default granular scoping for users. This depends on the structure and divisions within your organization and the particular purpose of each organizational unit to which scoped users belong.

1. Ensure that you have the necessary administrator-level permissions.
2. Verify that the users, user groups, and API keys defined in Cortex AgentIX are assigned the relevant scopes.
  - To verify the granular scoping of a user, select Settings → Configurations → Access Management → Users, right-click the user name, and select Edit User Permissions.
  - To verify the granular scoping of a user group, select Settings → Configurations → Access Management → User Groups, right-click the user group, and select Edit Group.
  - To verify the granular scoping of an API key, select Settings → Configurations → Integrations → API Keys, right-click the API key, and select Edit.
3. In the Scope tab, expand the scoping areas to review the current granular scoping definitions by clicking the chevron icon (>) beside the scoping area title, and make any changes required. The following table explains the options available to configure:

**IMPORTANT:**

Before configuring, ensure that you review the Understand scoping section.

Scoping Area	Granular Scoping Configurations
Assets	<p>Set the Scope by selecting one of the following:</p> <ul style="list-style-type: none"><li>• No assets: No asset is accessible.</li><li>• All assets: Defines access to all assets.</li><li>• Select asset groups: Defines access to the specific assets associated with the Asset Groups selected, and to view all their related cases, issues, and findings for these specific assets and Asset Groups. Under Select asset groups, define the specific asset groups that you want to grant access. Only Asset Groups relevant for scoping are listed, which are asset groups that are using only the asset attributes listed in Manage user scope (under Understand scoping → Scoping Areas → Assets).</li></ul> <p>The scoping of assets also affects the scoping of cases, issues, and findings.</p> <p><b>NOTE:</b></p> <p>Visibility of Security domain Issues that refer to assets with agents is controlled by the Endpoints scoping configuration.</p>



Scoping Area	Granular Scoping Configurations
Cases and Issues	<p>Set the Scope by selecting one of the following:</p> <ul style="list-style-type: none"> <li>No cases and issues: Defines access to no cases and issues.</li> <li>All cases and issues: Defines access to all cases and issues. Users can view cases or issues referencing assets within their scope. Use the Assets section to define which assets are in scope.</li> <li>Select domains: Defines access to the domains selected to view their related cases and issues. Under Select domains, define the specific domains that you want to grant access.</li> </ul> <p>Users can only view cases or issues referencing assets and endpoints within their scope. Use the Assets section to define which assets are in scope.</p> <p>When selecting All cases and issues or Select domains, you can separately configure access to issues and cases that lack an asset reference or where the referenced asset is not in All Assets and All Endpoints inventories. To provide access, select the Allow access to cases and issues that are not referencing known assets or endpoints checkbox.</p>
Endpoints	<p>Set the Scope by selecting one of the following:</p> <ul style="list-style-type: none"> <li>No endpoints: Defines access to no endpoints with no ability to view their related agent management and enterprise policies.</li> <li>All endpoints: Defines access to all endpoints with the ability to view their related agent management and enterprise policies. This configuration can impact the visibility of related Security domain Cases and Issues, but will not affect asset visibility.</li> <li>Select specific (at least one required): Defines specific access to all endpoint groups by selecting Endpoint Groups or all endpoint tags by selecting Endpoint Tags to view their related agent management and enterprise policies. This configuration can impact the visibility of related Security domain Cases and Issues, but will not affect asset visibility.</li> </ul>

4. Click Save.

5. Repeat steps 2 to 4 until you have configured all users, user groups, and API keys with the correct granular scoping access.

6. Enable granular scoping in Cortex AgentiX.

a. Select Settings → Configurations → General → Server Settings, and select the Enable Scope Based Access Control toggle.

b. (Optional) You can select the Endpoint Scoping Mode, which is defined per tenant:

- Permissive: Enables users with at least one scope tag to access the relevant entity with that same tag.
- Restrictive: Users must have all the scoped tags that are tagged within the relevant entity of the system.

c. Click Save.

When you are finished, all the users in Cortex AgentiX are now able to use Cortex AgentiX only within the granular scoping granted according to their assigned user roles.

### 3.5.5 | Manage access to objects

#### Abstract

Learn more about managing access to objects in Cortex AgentiX.

Cortex AgentiX enforces least-privileged access by allowing you to manage access for individual instances of custom (user-defined) and system Dashboards and Saved Queries. Access management for these items is handled through a common experience for per-object access, which allows you to treat these tools as distinct objects with their own access settings.

- Custom objects:** User-defined objects that can be fully managed, shared, or deleted by the Owner or an authorized Editor.
- System objects:** Out-of-the-box objects provided by Palo Alto Networks. These are available to any user with access to Dashboards or to Saved Queries and cannot be deleted or have their ownership changed, though they can often be duplicated to create a custom version.

#### Key concepts

Before configuring access, it is important to understand the different states and roles that define an object's security access.

#### General access states



The General access setting determines the baseline visibility for an object:

- **Restricted** (default): The object is visible only to the Owner and those specifically shared with.
- **Public**: The object is visible to all users who have that component enabled in their role permissions. Any user with access to the component can view both Public and System objects, and those with the required role permissions can also edit the Public custom objects.

#### Per-object roles

- **Owner**: The person who created the object. Every object has an assigned Owner responsible for managing its lifecycle and access. Owners have full control, including the ability to edit content, delete the object, and, depending on tenant-level settings, share the object with other principals (users, user groups, or API keys) as an Editor or Viewer.
- **Editor**: Can view and modify the object. If authorized by tenant-level settings, they can also manage access for others.
- **Viewer**: Can view the object and its data but cannot make any changes to the object's configuration or access settings.

**Administrative access**: Account and Instance Administrators have inherent visibility into all objects (including Restricted ones) regardless of whether they have been explicitly shared with them. They can also Change Owner for any object.

Keep in mind the following

While Per-object access controls the visibility of the dashboard or saved query, the underlying data remains governed by Scope-Based Access Control (SBAC). A user must have the appropriate SBAC permissions to view the data available through an object.

#### Sharing icons

The following icons indicate the sharing status and origin of an object in management tables:

- : A Restricted object you created that is not shared with anyone else.
- : An object you created that is currently shared with other users, groups, or API keys.
- : An object created by another user that has been shared with you.
- : A Palo Alto Networks object provided out-of-the-box. These are Public, cannot be deleted, and ownership cannot be transferred.

#### How to configure access to objects?

Configuring access follows a top-down workflow:

1. Tenant-level settings: Establish the "rules of engagement" for the entire instance.
2. Role permissions: Enable specific components and define additional capabilities for those roles.
3. Per-object access: Manage visibility and access levels for specific dashboards and queries.
4. Scope-Based Access Control (SBAC): Ensure the user has the required permissions to view the underlying data available through the object.

#### Step 1: Configure tenant-level access settings

Administrators first establish the "rules of engagement" for all objects. These settings are located under Settings → Configurations → Access Management → Objects:

- Owners can Share objects they created: Allows the creator (Owner) of an object to share it with users, user groups, or API keys.
  - Editors can also Share objects with others: Allows users with Editor access to further share the object with additional principals (users, user groups, and API keys).
- Owners and editors can change the general access (default): Allows the object owner and any user with Editor access to modify the object's General access settings (Restricted or Public) using the drop-down menu in the object's sharing settings.

#### Step 2: Set role permissions

Once tenant-level policies are established, configure individual roles to allow users to interact with specific components:

1. Select Settings → Configurations → Access Management → Roles.
2. Right-click the relevant user role, and select Edit Role.
3. Under Components, expand each list, set the applicable component to one of the following:
  - **Disabled**: The component is hidden from the user's navigation menu. The user cannot access any objects associated with this component, even if they were previously shared with them.
  - **Enabled**: The component is visible in the user's navigation menu. The user can view Public objects and any Restricted objects shared with them.
4. Define additional capabilities.

If enabled, refine capabilities using the following checkboxes:



- Create [Object]: Allows the user to create new instances; the user is automatically designated as the Owner, which grants the inherent right to edit, delete, and manage sharing for that specific object.
- Edit Public [Object]: Allows the user to modify custom objects that have been set to Public General access, even if they are not the owner.

Once a component is enabled using role permissions, sharing is managed at the individual object level. Owners and authorized editors can share with other principals (users, user groups, or API keys) directly on the object.

#### Step 3. Configuring per-object access

For more information on managing visibility and access levels for specific dashboards and saved queries, see the following topics:

- Manage access to custom dashboards
- Manage access to saved queries

#### Step 4. Configure SBAC permissions

For more information on managing user scope so users have the permissions necessary to view the data available through the object, see Manage user scope.

How to change an object owner

To ensure continuity when personnel changes occur or a user leaves the organization, the ownership of an object (a dashboard or a saved query) can be changed.

- **Administrative privilege:** Only Account and Instance Administrators can change the owner of an object. Other users who are Owners and Editors cannot perform this action.
- **Change Owner:** Using the Change Owner action in the management table of the specific object, administrators can select a new user to take over full control. Once changed, the new user assumes all Owner-level rights, including the ability to edit, delete, and share with other principals (users, user groups, and API keys).

Access examples

Granular per-object access supports various organizational security requirements:

1. **Use only by SOC team:** A "flat" structure where all analysts can see all objects. This is the default setting for the tenant. By default, newly created custom objects, such as a specific investigation dashboard or a complex XQL saved query, are Restricted and visible only to the creator; the owner can then make them Public to allow the entire team to view or edit them based on their role permissions.
2. **Both SOC team and Internal threat:** Specific objects, such as sensitive dashboards and saved queries, are created by a member of the Internal Threat team and made accessible only to the Internal Threat user group. First, an administrator must enable the tenant-level access settings that allow users to share objects. Members of the Internal Threat team then create these objects and share them only with their peers or their specific user group. Members of the SOC team do not have access to these dashboards and saved queries, as they are not visible or accessible to any users who have not been explicitly granted access.
3. **Both SOC team and Cloud team:** Provides department isolation. Each team only accesses its own saved queries and dashboards; the SOC team cannot see Cloud team objects, and vice versa.

##### 3.5.5.1 | Manage access to custom dashboards

Abstract

Learn more about managing access to custom dashboards in Cortex AgentIX.

The Dashboard Manager serves as the central repository for your visualizations. By using object-level access, you can ensure that custom (user-defined) dashboards, such as those used for sensitive executive reporting or specialized department views, are only accessible to authorized users and user groups. The permissions assigned to your role, combined with the ownership of specific objects, directly determine the content available to you; you can only access dashboards where you are the Owner, dashboards that have been explicitly shared with you (or your user group), or dashboards marked as Public.

#### PREREQUISITE:

- **Configure tenant-level settings:** An administrator must first establish the sharing framework under Settings → Configurations → Access Management → Objects.

The configuration of these settings defines the authorized sharing workflows for custom dashboards:

- **Enable "Owners can Share objects they created":** Grants owners the ability to share dashboards with specific users and user groups. In the Dashboard Manager, this enables the Share option.
- **Disable "Owners can Share objects they created":** Restricts owners to managing only General access (Public vs. Restricted). In the Dashboard Manager, this replaces the Share option with the Manage Access option.
- **Define Scope-Based Access Control (SBAC):** While object-level sharing grants access to the dashboard's layout and configuration, users must also have the appropriate SBAC permissions to view the actual data populated within the widgets. If a user has access to a shared dashboard but lacks the required data scope for the underlying datasets, the dashboard will load, but the widgets may appear empty or display an error.

For more information on these prerequisites, see Manage access to objects.

Understanding widget behavior



Because dashboards are composed of multiple widgets, it is important to understand how access is applied to these individual components:

- **Widgets are not objects:** Unlike dashboards, individual widgets are not treated as independent objects. They do not have their own "Share" dialog and cannot be shared independently. Within the Widget Library, a widget is set to either Restricted (visible only to the creator) or Public (visible to all with Widget Library access).
- **Inherited access:** Any user who has been granted access to a custom dashboard (as a Viewer or Editor) can see all the widgets contained within that dashboard, including those marked as Restricted. This means you may see a widget on a shared dashboard that you cannot see in the Widget Library even if you have access to it.
  - **Dashboard Editors:** Can edit the dashboard layout, but the widget is only available in their Widget Library for editing when the widget is Public.
  - **Dashboard Viewers:** Can't make any changes to dashboards or widgets that are Restricted.

How to configure access to custom dashboards

Step 1: Set role-level permissions

Role permissions define the functional capabilities for dashboards and the Widget Library, and determine what actions a user can take.

1. Select Settings → Configurations → Access Management → Roles.
2. Right-click the relevant user role, and select Edit Role.
3. Under Components, expand Dashboards & Reports, and locate Dashboards.
4. Configure access state:
  - Disabled: Users cannot navigate to Dashboards & Reports → Dashboard Manager or Dashboards & Reports → Widget Library. Dashboards cannot be shared with this role. If the user previously owned or had access to shared dashboards, they are no longer available.
  - Enabled: Allows dashboards to be accessed and managed according to defined sub-permissions. Grants access to the Widget Library as explained below in Manage the Widget Library.
5. If Enabled, assign specific capabilities to control the UI:
  - Create Dashboards: Enables the New Dashboard button on the Dashboard Manager page, allowing the user to create new custom dashboard objects. The user who performs this action becomes the Owner of the object and is granted the inherent right to edit, delete, and manage sharing for that specific object.
  - Edit Public Dashboards: Allows the user to modify custom dashboards set to Public, even if they are not the owner.
6. Click Save.

Step 2: Manage the widget library

The Widget Library is the central repository for predefined and custom widgets and is intended for browsing and selecting widgets to add to a dashboard. Access to and visibility within the Widget Library is determined by role-level permissions and your specific access level to the dashboards where those widgets reside:

- **Access to the Widget Library:** To access the Widget Library, your role must have the Create Dashboards or Edit Public Dashboards capability. Users who only have "View" permissions for dashboards cannot access the Widget Library.
- **Widget Library visibility:** Visibility within the Widget library depends on ownership and inherited dashboard and widget permissions:
  - **Public and personal widgets:** You can always see widgets you created (Restricted) and widgets marked as Public.
  - **Inherited access via dashboards:** If a Restricted widget was created by another user but is part of a dashboard shared with you, you won't see it in the Widget Library and it can't be edited (unless you are an administrator).

#### **NOTE:**

If you're designated as an Editor, you can always duplicate the widget and make your changes on the copy.

Step 3: Manage sharing for a custom dashboard

Once a custom dashboard exists in the Dashboard Manager, the Owner (or an authorized Editor) defines who can see or edit it.

1. Select Dashboards & Reports → Dashboard Manager.
2. Locate the custom dashboard that you want to share in the table.
3. Right-click the custom dashboard and select the available access option. The menu option you see depends on your tenant-level settings:
  - Share: Use this if your admin enabled sharing. It allows you to grant access to specific users/groups and change the General access (Public/Restricted).
  - Manage Access: Use this if sharing is disabled. It is a restricted view that only allows you to toggle the General access between Public and Restricted. You cannot grant access to specific individuals.
4. (If sharing is enabled) Search for the User or User Group, and assign the access level: Viewer (read-only) or Editor (can modify and share).



5. Set the General access state:

- Restricted: Private to the Owner and the others granted access.
- Public: Visible to all users with the Dashboard component enabled in their role.

6. Click Save.

Sharing icons in the Dashboard Manager

The following icons help you identify the security access of your custom dashboards:

- A Restricted custom dashboard you created that is not shared with anyone else.
- A custom dashboard you created that is currently shared with other users or user groups.
- A custom dashboard created by another user that has been shared with you.
- A standard system dashboard provided by Palo Alto Networks. These are always Public and cannot be deleted or edited, and their ownership cannot be transferred. Yes, you can Duplicate a system dashboard to create a custom version that you can then modify and share.

Change owner of a dashboard

To ensure continuity when personnel changes occur or to hand off management of a resource, only administrators can change the ownership of a custom dashboard.

#### **NOTE:**

Only Account Administrators and Instance Administrators have the authority to change the owner of an object.

1. Select Dashboards & Reports → Dashboard Manager.
2. Right-click the custom dashboard in the table and select Change owner.
3. Select the new owner from the list of users, and click Change.

#### 3.5.5.2 | Manage access to saved queries

Abstract

Learn more about managing access to saved queries in Cortex AgentiX.

Review the following:

- Manage access to objects

The Query Library serves as the central repository for your team's investigation logic. By using object-level access, you can ensure that specific Cortex Query Language (XQL) queries, such as those used for sensitive internal investigations or executive reporting, are only accessible to authorized users, user groups, and API keys.

#### **PREREQUISITE:**

**Configure tenant-level settings:** An administrator must first establish the sharing framework under Settings → Configurations → Access Management → Objects.

The configuration of these settings defines the authorized sharing workflows for saved queries in the Query Library, including the options that appear to users when clicking the three dot, vertical ellipsis (⋮) for a query in the Query Library:

- **Enable "Owners can Share objects they created":** Grants owners the ability to share saved queries with specific users, user groups, and API keys to the query's access list. In the Query Library, this enables the Share option.
- **Disable "Owners can Share objects they created":** Restricts owners to managing only General access (Public vs. Restricted). In the Query Library, this replaces the Share option with the Manage Access option.

For more information on these tenant-level configurations, see Manage access to objects.

How access impacts the Query Builder

The permissions assigned to your role, combined with the ownership of specific objects, directly change the tools available to you while working in the Query Builder:

- **Restricted versus Public visibility:** Your Query Library view is personalized. You will only see queries where you are the Owner, queries that have been explicitly shared with you (or your user group or API key), or queries marked as Public.
- **Context-sensitive functionality:** The permissions assigned to your role, combined with the ownership of specific objects, directly change the tools available to you while working in the Query Builder and the Query Library. UI elements like the Save as menu or the Share action only appear if you have the required functional capabilities.



## How to configure access to saved queries

Setting up access involves a two-part process: enabling the user interface (UI) elements in the role settings, and then defining the audience for individual saved query objects.

### Step 1: Define role capabilities

Role-level permissions act as the "master switch" for Query Builder functionality and determine what actions a user can take.

1. Select Settings → Configurations → Access Management → Roles.

2. Right-click the relevant user role, and select Edit Role.

3. Under Components, expand Investigation & Response.

4. Ensure Query Library is set to Enabled.

5. Define functional capabilities to control the UI:

- Create Queries: Selecting this enables the Save as drop-down menu in the Query Builder. This allows users to select Save as → Query to Library or Save as → Widget to Library. The user who performs this action becomes the Owner of the object and is granted the inherent right to edit, delete, and manage sharing for that specific object..
- Edit Public Queries: This allows a user to modify queries marked as Public by others.

#### NOTE:

If the role of a user is set to Edit Public Queries but not Create Queries, they can update existing public queries, but the Save as drop-down menu will be hidden, preventing them from creating new Query Library entries.

### Step 2: Manage sharing for a specific query

Once a query exists in the Query Library, the Owner (or an authorized Editor) can define who has permission to view (and run) or edit it.

1. Select Investigation & Response → Search → Query Builder → XQL.

2. Under the Query Library tab, locate the query that you want to share in the table.

3. Click the three dot, vertical ellipsis (⋮) and select the available action:

- Share: This option appears when Owners can Share objects they created is enabled in tenant-level settings. It allows you to manage both General access and specific principals (users, user groups, and API keys).
- Manage Access: This option appears when Owners can Share objects they created is disabled. It only allows you to change the General access state.

4. (If sharing is enabled) To share with specific entities (for Restricted queries):

- Search for the User, User Group, or API Key.
- Assign the access level: Viewer (can run/view) or Editor (can modify and, if permitted by tenant-level settings, share).

5. Set the General access drop-down menu (if authorized by tenant-level settings):

- Restricted: The query is private. It is only visible to the Owner and the specific principals added to the list.
- Public: The query is visible to every user who has the Query Library enabled in their role.

#### NOTE:

When the tenant-level setting Owners and editors can change the general access is unselected, the drop-down is disabled and only an administrator can configure this option.

6. Click Save.

### Sharing icons in the Query Library

The following icons in the Query Library table help you identify the security access of your queries:

- A Restricted query you created that is not shared with anyone else.
- A query you created that is currently shared with other users, user groups, or API keys.
- A query created by another user that has been shared with you.
- A standard system query provided by Palo Alto Networks. These are always Public and can't be deleted, or have their ownership transferred.



### 3.5.6 | Set up authentication

#### Abstract

Authenticate Cortex AgentiX users using SAML 2.0 or Cortex Gateway.

You can create users in the Customer Support Portal or by using SAML Single Sign-On (SSO) in the tenant. Users authenticate by doing the following:

- Authenticate through the Customer Support Portal

When users log into Cortex Gateway or the tenant (provided they are assigned a role) they are prompted to sign into the Customer Support Portal using their username and password or 2FA (if set up). This is the default method of authentication.

After you have created users, add them to user groups or assign roles directly.

- Authenticate using SAML single sign-on in the Cortex AgentiX tenant

Users can be authenticated using your IdP provider such as Okta, Ping, or Azure AD. You can use any IdP that supports SAML 2.0. After you configure the SSO integration you need to map group SAML group membership to user groups in Cortex AgentiX.

SSO authentication has the following advantages:

- Removes the administrative burden of requiring separate accounts to be configured through the Customer Support Portal.
- Enforces multi-factor authentication (MFA) and any conditional access policies on the user login at the IdP before granting a user access to Cortex AgentiX.
- Maps SAML group memberships to user groups and roles, allowing you to manage role-based access control.
- Removes access to Cortex AgentiX when a user is removed or disabled in the IdP.

Customer Support Portal authentication, by contrast, is useful if you have users who need the same permissions across multiple tenants. If you use SSO for multiple tenants, you must set up the SSO configuration separately for each tenant, both in the IdP and in Cortex AgentiX.

If you want to restrict the user login through SSO only, remove any direct role and user group mapping for the user on Cortex Gateway or the Cortex AgentiX tenant. This removes Customer Support Portal access for the user. You then need to ensure that you add the SAML group mapping. The user can access and acquire the user group and roles based on SAML group mapping. Once completed, the user is able to access Cortex AgentiX using SSO only and will not be able to use Customer Support Portal login method.

#### TIP:

You should have at least one user in the Customer Support Portal for backup, in case of any authentication issues with your IdP provider.

#### 3.5.6.1 | Authenticate users through the Customer Support Portal

#### Abstract

Authenticate Cortex AgentiX users when using the Customer Support Portal.

When you add users to your Customer Support Portal account, users are sent an invitation to join. After they accept, users can access Cortex Gateway and tenants, but they cannot view any tenants in the Gateway and cannot view any data in the tenant unless they are assigned a direct role or user group role. Only Account Admins can make any changes in Cortex Gateway.

#### NOTE:

You must be assigned the Super User role in the Customer Support Portal to add users in the Customer Support Portal.

The first Super User who logs into Cortex Gateway is automatically assigned the Account Admin role and has access to the tenant. The user who activates the Cortex AgentiX tenant will also be assigned the Account Admin role (if there is no current Account Admin role) or Instance Admin (if there is an existing Account Admin role) and will have access to the tenant. Any additional users including Super Users need to be assigned access to the tenant.

When users log into Cortex Gateway or the tenant they are prompted to sign into the Customer Support Portal using their username and password. This is the default method of authentication.

#### NOTE:

After users are added to the Customer Support Portal and they accept the invitation, you can manage them in Cortex Gateway or the Cortex AgentiX tenant.

How to authenticate users through the Customer Support Portal

1. Add users to your Customer Support Portal account, by logging into <https://support.paloaltonetworks.com/> and doing one of the following:



- In your Customer Support User Account, create users.
  1. On the left-hand side menu, select Members → Create New User .
  2. Add the member details and click Submit.

An email is sent to the user which must be accepted within seven days.

For more detailed information including how to reset the invitation, see [How a Super User Creates a New Customer Support Portal User Account](#).

  - Send an Account Registration Link.

A registration link is generated by a Customer Support Portal account Super User and shared with users who need to create a login for access to the account.

  1. On the left-hand side menu, select Account Management → Account Details, and click User Access.
  2. In the Account Registration link, click Create.
  3. Copy and send the link to the users you want to add.

When clicking the link, users are required to enter their registration details and submit them to the Customer Support Portal.

After users have submitted their details, the Super User receives a notification that a user has been created.

For more information about how to generate, regenerate, or disable a link, see [How to Use the Account Registration Link](#).

## 2. Log in to Cortex Gateway .

After the user accepts the invitation, you see the added users. You must assign a role to the user directly or add them to user groups in Cortex Gateway or in the Cortex AgentiX tenant.

### 3.5.6.2 | Authenticate users using SSO

#### Abstract

Set up authentication in the Cortex AgentiX tenant using SSO.

Cortex AgentiX enables you to authenticate system users securely across enterprise-wide applications and websites with one set of credentials using single sign-on (SSO) with SAML 2.0. System users can authenticate using your organization's Identity Provider (IdP), such as Okta or PingOne. You can integrate with any IdP that is supported by SAML 2.0.

Configuring SSO with SAML 2.0 is dependent on your organization's IdP. Some of the parameter values need to be supplied from your organization's IdP and some need to be added to your organization's IdP. You must have sufficient knowledge about IdPs, how to access your organization's IdP, which values to add to Cortex AgentiX, and which values to add to your IdP fields.

#### NOTE:

- To set up SSO authentication in the tenant, you must be assigned an Instance Administrator or Account Admin role.
- SAML 2.0 users must log in to Cortex AgentiX using the FQDN (full URL) of the tenant. To allow login directly from the IdP to , you must set the relay state on the IdP to the FQDN of the tenant.
- If you have multiple tenants, you must set up the SSO configuration separately for each tenant, both in the IdP and in Cortex AgentiX.
- Create groups in Cortex AgentiX that correspond to the groups in your IDP. Add the appropriate SAML group mapping from your IdP to each of the user groups in Cortex AgentiX.
- When a user logs in for the first time, the user account is automatically created (JIT provisioning), provided you mapped groups. This process requires either using the default role option or ensuring that SSO groups are properly mapped to Cortex AgentiX groups. If the user belongs to a group that has a mapping, the user will be granted access automatically upon login.
- If you are using AWS SSO, the Application ACS URL refers to the Single Sign-On URL and the Application SAML Audience refers to the Audience URL (SP Entity ID). Both values can be copied from the Authentication Settings in Cortex AgentiX.

If you are configuring Okta or Azure, follow the procedure in Okta or Azure AD. You can also adapt these instructions for use with any similar SAML 2.0 IdP.

1. If you want to add another SSO connection to enable managing user groups with different roles and different IdPs, click Add SSO Connection.

Different SSO parameters for an SSO are displayed to configure according to your organization's additional IdP.

#### NOTE:



- The first SSO cannot be deleted, it can only be deactivated by toggling SSO Enabled to off.
  - The Domain parameter is predefined for the first SSO.
- If you add additional SSO providers, you must provide the email Domain in the SSO Integration settings for all providers except the first. Cortex AgentiX uses this domain to determine to which identity provider to send the user for authentication.
- When mapping IdP user groups to Cortex AgentiX user groups, you must include the group attribute for each IdP you want to use. For example, if you are using Microsoft Azure and Okta, your Cortex AgentiX user group SAML Group Mapping field must include the IdP groups for each provider. Each group name is separated by a comma.

2. Set the following parameters using your organization's IdP.

- General parameters
- IdP Attribute Mapping
- Advanced Settings (optional)

3. Save your changes.

Whenever an SSO user logs in to Cortex AgentiX, the following login options are available.

- Sign-in with SSO

If you have enabled more than one SSO provider, an optional email field appears. If the user does not enter an email address or if the email address does not match an existing domain, the user is automatically directed to the default IdP provider (the first in the list of SSO providers in the Authentication Settings). If the user enters an email address and it matches a domain listed in the Domain field in the SSO Integration settings for one of your IdPs, Sign-In with SSO sends the user to the IdP associated with that email domain.

#### General parameters

Parameter	Description
IdP SSO or Metadata URL	Select the option that meets your organization's requirements.  Indicates your SSO URL, which is a fixed, read-only value based on your tenant's URL using the format <code>https://&lt;name of tenant&gt;.crtx.paloaltonetworks.com/idp/saml</code> . For example, <code>https://tenant1.crtx.paloaltonetworks.com/idp/saml</code>  You need this value when configuring your IdP.
IdP SSO URL	Specify your organization's SSO URL, which is copied from your organization's IdP.
Metadata URL	
Audience URI (SP Entity ID)	Indicates your Service Provider Entity ID, also known as the ACS URL. It is a fixed, read-only value using the format, <code>https://&lt;name of tenant&gt;.paloaltonetworks.com</code> . For example <code>https://tenant1.crtx.paloaltonetworks.com</code> .  You need this value when configuring your organization's IdP.
Default Role	(Optional) Select the default role that you want any user to automatically receive when they are granted access to Cortex AgentiX through SSO. This is an inherited role and is not the same as a direct role assigned to the user.
IdP Issuer ID	Specify your organization's IdP Issuer ID, which is copied from your organization's IdP.
X.509 Certificate	Specify your X.509 digital certificate, which is copied from your organization's IdP.



Parameter	Description
Domain	Relevant only for multiple SSOs. For one SSO, this is a fixed, read-only value. Associate this IdP with a specific email domain (user@<domain>). When logging in, users are redirected to the IdP associated with their email domain or to the default IdP if no association exists.

#### IdP attribute mapping

These IdP attribute mappings are dependent on your organization's IdP.

Parameter	Description
Email	Specify the email mapping according to your organization's IdP.
Group Membership	Specify the group membership mapping according to your organization's IdP. <b>NOTE:</b> Cortex AgentiX requires the IdP to send the group membership as part of the SAML token. Some IdPs send values in a format that include a comma, which is not compatible with Cortex AgentiX. In that case, you must configure your IdP to send a single value without a comma for each group membership. For example, if your IdP sends the Group DN (a comma-separated list), by default, you must configure IdP to send the Group CN (Common Name) instead.
First Name	Specify the first name mapping according to your organization's IdP.
Last Name	Specify the last name mapping according to your organization's IdP.

#### Advanced settings

The following advanced settings are optional to configure and some are specific for a particular IdP.

Parameter	Description
Relay State	(Optional) Specify the URL for a specific page that you want users to be directed to after they've been authenticated by your organization's IdP and log in to Cortex AgentiX.
IdP Single logout URL	(Optional) Specify your IdP single logout URL provided by your organization's IdP to ensure that when a user initiates a logout from Cortex AgentiX, the identity provider logs the user out of all applications in the current identity provider login session.
SP Logout URL	(Optional) Indicates the Service Provider logout URL that you need to provide when configuring a single logout from your organization's IdP to ensure that when a user initiates a logout from Cortex AgentiX, the identity provider logs the user out of all applications in the current identity provider login session. This field is read-only and uses the following format <a href="https://&lt;name of tenant&gt;.crtx.paloaltonetworks.com/idp/logout">https://&lt;name of tenant&gt;.crtx.paloaltonetworks.com/idp/logout</a> , such as <a href="https://tenant1.crtx.paloaltonetworks.com/idp/logout">https://tenant1.crtx.paloaltonetworks.com/idp/logout</a> .
Service Provider Public Certificate	(Optional) Specify your organization's IdP service provider public certificate.



Parameter	Description
Service Provider Private Key (Pem Format)	(Optional) Specify your organization's IdP service provider private key in Pem Format.
Remove SAML RequestedAuthnContext	<p>(Optional) Requires users to log in to Cortex AgentiX using additional authentication methods, such as biometric authentication.</p> <p>Selecting this removes the error generated when the authentication method used for previous authentication is different from the one currently being requested. See here for more details about the <code>RequestedAuthnContext</code> authentication mismatch error.</p>
Force Authentication	(Optional) Requires users to reauthenticate to access the Cortex AgentiX tenant if requested by the idP, even if they already authenticated to access other applications.

#### Troubleshoot SSO issues

The following list describes the common errors and issues when using SAML 2.0 authentication.

- Errors in your IdP could mean the Service Provider Entity ID and/or Service Identifier are not properly configured in the IdP or in the Cortex AgentiX settings.
- SAML attributes from the IdP are not properly mapped in Cortex AgentiX. The attributes are case sensitive and must exactly match in your IdP and in the Cortex AgentiX IdP Attributes Mapping.
- Group memberships from the IdP have not been properly mapped to Cortex AgentiX user groups. Verify the values your identity provider is sending, to properly map the groups in Cortex AgentiX.
- The identity provider is not configured to sign both the SAML response and the assertion on the login token. Your IdP must be configured to sign both to ensure a secure login.
- If you require further troubleshooting, we recommend using your browser's built-in developer tools or additional browser plugins to capture the login request and SAML token.

#### 3.5.6.3 | Set up Okta as the Identity Provider Using SAML 2.0

This topic provides specific instructions for using Okta to authenticate your Cortex AgentiX users. As Okta is a third-party software, specific procedures, and screenshots may change without notice. We encourage you to also review the Okta documentation for app integrations.

To configure SAML SSO in Cortex AgentiX, you must be a user who can access the Cortex AgentiX tenant and have either the Account Admin or Instance Administrator role assigned.

##### Task 1. Configure Okta Groups

Within Okta, assign users to groups that match the user groups they will belong to in Cortex AgentiX. Users can be assigned to multiple Okta groups and receive permissions associated with multiple user groups in Cortex AgentiX. Use an identifying word or phrase, such as Cortex AgentiX, within the group names. For example, Cortex AgentiX Analysts. This allows you to send only relevant group information to Cortex AgentiX, based on a filter you will set in the group attribute statement.

Create a list of the Okta groups and their corresponding Cortex AgentiX user groups (or the Cortex AgentiX user groups you intend to create) and save this list for later use when configuring user groups in Cortex AgentiX.

##### Task 2. Copy Single SSO and Audience URI Values from Cortex AgentiX

1. Expand the SSO Integration settings.
2. Copy and save the values for Single Sign-On URL and Audience URI (SP Entity ID).

Both values are needed to configure your IdP settings.

You cannot save the enabled SSO Integration at this time, as it requires values from your IdP.

##### Task 3. Configure Cortex AgentiX Application in Okta



1. In Okta, create a Cortex AgentiX application and Edit the SAML Settings.
2. Paste the Single sign-on URL and the Audience URI (SP Entity ID) that you copied from the Cortex AgentiX SSO settings. The Audience URI should also be pasted in the Default RelayState field, which allows users to log in to Cortex AgentiX directly from the Okta dashboard.
3. Click Show Advanced Settings, verify that Okta is configured to sign both the response and the assertion signature for the SAML token, and then click Hide Advanced Settings.
4. Cortex AgentiX requires the IdP to send four attributes in the SAML token for the authenticating user.
  - Email address
  - Group membership
  - First Name
  - Last Name

Configure Okta to send group memberships of the users using the `memberOf` attribute. Use the word or phrase you selected when configuring Okta groups (such as Cortex AgentiX) to create a filter for the relevant groups.

5. Copy the exact names of the attribute statements from Okta and save them, as they are required to configure the Cortex AgentiX SSO integration. In the example above, the names are FirstName, LastName, Email, and memberOf. The attribute names are case-sensitive.

#### Task 4. Copy IdP SSO URL, Identity Provider Issuer, and X.509 Certificate Values

1. In Okta, from your Cortex AgentiX application page, click View SAML setup instructions. If you do not see this button, verify you are on the Sign On tab of the application.
2. Copy and save the values for Identity Provider Single Sign-On URL, Identity Provider Issuer, and the X.509 Certificate. These values are needed to configure your Cortex AgentiX SSO Integration.

#### Task 5. Configure the Cortex AgentiX SSO Integration

1. Expand the SSO Integration settings.
2. Use the following table to complete the SSO Integration settings, based on the values you saved from Okta.

Okta	Cortex AgentiX Field
Identity Provider Single Sign-On URL	IdP SSO URL
Identity Provider Issuer	IdP Issuer ID
X.509 Certificate	X.509 Certificate

3. In the IdP Attributes Mapping section, enter the attribute names from Okta. The names are case-sensitive and must match exactly.
4. Save your settings.

#### Task 6. Map SAML Group Memberships to Cortex AgentiX User Groups

1. Right-click a user group and select Edit Group.
2. In the SAML Group Mapping field add the Okta group(s) that should be associated with this user group. Multiple groups should be separated with a comma. The Okta group name must match the exact value sent in the token.
3. Save your settings.
4. Repeat for each user group.

#### Task 7. Test SSO Login

1. Go to the Cortex AgentiX tenant URL and Sign-In with SSO.

#### NOTE:

When using SAML 2.0, users are required to authenticate by logging in directly at the tenant URL. They cannot log in via Cortex Gateway.

2. After authentication to Okta, you are redirected again to the Cortex AgentiX tenant.



3. When logged in, validate that you have been assigned the proper roles.

To view your role and any role assigned to a user group you are a member of, click your name in the bottom left-hand corner, and click About.

#### 3.5.6.4 | Set up Azure AD as the Identity Provider Using SAML 2.0

This topic provides specific instructions for using Azure AD to authenticate your Cortex AgentiX users. As Azure AD is a third-party software, specific procedures, and screenshots may change without notice. We encourage you to also review the Azure AD documentation.

To configure SAML SSO in Cortex AgentiX, you must be a user who can access the Cortex AgentiX tenant and have either the Account Admin or Instance Administrator role assigned.

The following video is a step-by-step guide configuring SSO for Azure AD: Azure AD SSO.

##### Task 1. Configure Azure AD Security Groups

Within Azure AD, assign users to security groups that match the user groups they will belong to in Cortex AgentiX. Users can be assigned to multiple Azure AD groups and receive permissions associated with multiple user groups in Cortex AgentiX. Use an identifying word or phrase, such as Cortex AgentiX, within the group names. For example, Cortex AgentiX Analysts. This allows you to send only relevant group information to Cortex AgentiX, based on a filter you will set in the group attribute statement.

##### Task 2. Copy Single SSO and Audience URI Values from Cortex AgentiX

1. By default, SSO is disabled in Cortex AgentiX.
2. Expand the SSO Integration settings.
3. Copy and save the values for Single Sign-On URL and Audience URI (SP Entity ID).

Both values are needed to configure your IdP settings.

##### **IMPORTANT:**

When copying the Single Sign-On URL value, remove `idp/saml` and leave the trailing `/`.

For example, if the Single Sign-On URL is `https://clientname.panproduct.region.paloaltonetworks.com/idp/saml`, just copy `https://clientname.panproduct.region.paloaltonetworks.com/`.

4. You cannot save the enabled SSO Integration at this time, as it requires values from your IdP.

##### Task 3. Configure Cortex AgentiX Application in Azure AD

1. From within Azure AD, create a Cortex AgentiX application and Edit the Basic SAML Configuration.



## Set up Single Sign-On with SAML

An SSO implementation based on federation protocols improves security, reliability, and end user experiences and is easier to implement. Choose SAML single sign-on whenever possible for existing applications that do not use OpenID Connect or OAuth. [Learn more.](#)

Read the [configuration guide](#) for help integrating Cortex XSOAR 8 Production.

The screenshot shows two configuration sections for SAML setup:

**1 Basic SAML Configuration**

Identifier (Entity ID)	<a href="https://trainingxsoar.xsoarbenignnetwork.com">https://trainingxsoar.xsoarbenignnetwork.com</a>
Reply URL (Assertion Consumer Service URL)	<a href="https://trainingxsoar.xsoarbenignnetwork.com/idp/saml">https://trainingxsoar.xsoarbenignnetwork.com/idp/saml</a>
Sign on URL	<a href="https://trainingxsoar.xsoarbenignnetwork.com">https://trainingxsoar.xsoarbenignnetwork.com</a>
Relay State (Optional)	<a href="https://trainingxsoar.xsoarbenignnetwork.com">https://trainingxsoar.xsoarbenignnetwork.com</a>
Logout Url (Optional)	Optional

**2 Attributes & Claims**

givenname	user.givenname
surname	user.surname
emailaddress	user.mail
name	user.userprincipalname
memberOf	user.groups
Unique User Identifier	user.userprincipalname

- Paste the Single sign-on URL and the Audience URI (SP Entity ID) that you copied from the Cortex AgentiX SSO settings. The Single sign-on URL from Cortex AgentiX should be pasted in the Reply URL and the Sign on URL fields. The Audience URI (SP Entity ID) value from Cortex AgentiX should be pasted in the Identifier (Entity ID) and Relay State fields. This allows users to log in to Cortex AgentiX directly from Azure AD.



## Basic SAML Configuration

Save | Got feedback?

**Identifier (Entity ID)** \* ⓘ  
The unique ID that identifies your application to Microsoft Entra ID. This value must be unique across all applications in your Microsoft Entra tenant. The default identifier will be the audience of the SAML response for IDP-initiated SSO.

Default
https://[REDACTED].com

Add identifier  
Patterns: https://samitoolkit.azurewebsites.net

**Reply URL (Assertion Consumer Service URL)** \* ⓘ  
The reply URL is where the application expects to receive the authentication token. This is also referred to as the "Assertion Consumer Service" (ACS) in SAML.

Index	Default
https://[REDACTED]	<input checked="" type="checkbox"/> ⓘ

Add reply URL  
Patterns: https://samitoolkit.azurewebsites.net/SAML/Consume

**Sign on URL** \* ⓘ  
Sign on URL is used if you would like to perform service provider-initiated single sign-on. This value is the sign-in page URL for your application. This field is unnecessary if you want to perform identity provider-initiated single sign-on.

https://[REDACTED].com/	✓
-------------------------	---

Patterns: https://samitoolkit.azurewebsites.net/

**Relay State (Optional)** ⓘ  
The Relay State instructs the application where to redirect users after authentication is completed, and the value is typically a URL or URL path that takes users to a specific location within the application.

https://[REDACTED].com/
-------------------------

3. In the SAML Certificates section, click Edit and verify that Azure is configured to sign both the response and the assertion.

## SAML Signing Certificate

Manage the certificate used by Azure AD to sign SAML tokens issued to your app

Save | New Certificate | Import Certificate | Got feedback?

Status	Expiration Date	Thumbprint
Active	8/30/2026, 4:03:21 PM	0120401318F5DC6F084CEFB1E70AD49FA97D276A

**Signing Option** Sign SAML response and assertion

**Signing Algorithm** SHA-256

4. To have Azure AD send group membership for the user in the SAML token, you must + Add a group claim in the Attributes & Claims section. Send the Security groups, using the source attribute Group ID. Use the word or phrase you selected when configuring Azure AD security groups (such as Cortex AgentX) to create a filter. Customize the name of the group claim as memberOf.

## Group Claims

Manage the group claims used by Azure AD to populate SAML tokens issued to your app

Which groups associated with the user should be returned in the claim?

None  
 All groups  
 Security groups  
 Directory roles  
 Groups assigned to the application

**Source attribute \***

Emit group name for cloud-only groups ⓘ

Advanced options

Filter groups

**Attribute to match \***

**Match with \***

**String \***

Customize the name of the group claim

**Name (required)**

**Namespace (optional)**

Emit groups as role claims ⓘ

Apply regex replace to groups claim content

5. In addition to group membership, verify that there are also claims for:

- Email address
- First Name
- Last Name

#### Task 4. Copy Login URL, Azure ID Identifier, and Attribute Claims

1. In Azure, from the Single sign-on page, in the Set up Cortex AgentiX Production section, copy the values for the Login URL and Azure AD Identifier. You need these values to configure the SSO Integration in Cortex AgentiX.

### Set up Cortex XSOAR 8 Production

You'll need to configure the application to link with Azure AD.

Login URL	<input type="text" value="https://agentix.cortexxsoar.com:443/agentix/auth/realms/cortexxsoar/protocol/openid-connect/auth?client_id=agentix&amp;response_type=code&amp;scope=openid+profile+email+offline_access&amp;state=676938000022525..."/>
Azure AD Identifier	<input type="text" value="https://agentix.cortexxsoar.com:443/agentix/auth/realms/cortexxsoar/protocol/openid-connect/issuer"/>
Logout URL	<input type="text" value="https://agentix.cortexxsoar.com:443/agentix/auth/realms/cortexxsoar/protocol/openid-connect/logout?post_logout_redirect_uri=https://agentix.cortexxsoar.com:443/agentix/auth/realms/cortexxsoar/protocol/openid-connect/auth?client_id=agentix&amp;response_type=code&amp;scope=openid+profile+email+offline_access&amp;state=676938000022525..."/>

2. Edit Attributes & Claims and copy the values in the Claim name column. The claim name is case sensitive. You need these values to configure the SSO Integration in Cortex AgentiX.



**NOTE:**

The default attributes shown on the main single sign-on page in Azure AD are not the values you need. You must click Edit next to Attributes and Claims to view and copy the actual values.

Required claim		
Claim name	Type	Value
Unique User Identifier (Name ID)	SAML	user.userprincipalname [...]
Additional claims		
Claim name	Type	Value
<a href="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name">http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name</a>	SAML	user.mail
<a href="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname">http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname</a>	SAML	user.givenname
<a href="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name">http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name</a>	SAML	user.userprincipalname
<a href="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname">http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname</a>	SAML	user.surname
memberOf	SAML	user.groups

## Task 5. Download the Certificate

From the SAML Certificates section in Azure AD, Download the Certificate (Base64). You need the contents of this file to configure the Cortex AgentiX SSO Integration.

SAML Certificates	
<b>Token signing certificate</b>	<a href="#">Edit</a>
Status	Active
Thumbprint	0120401318F5DC6F084CEFB1E70AD49FA97D276A
Expiration	8/30/2026, 4:03:21 PM
Notification Email	<a href="#">Send notification</a>
App Federation Metadata Url	<a href="https://login.microsoftonline.com/02525600-2525-4025-8025-02525600-2525">https://login.microsoftonline.com/02525600-2525-4025-8025-02525600-2525</a>
Certificate (Base64)	<a href="#">Download</a>
Certificate (Raw)	<a href="#">Download</a>
Federation Metadata XML	<a href="#">Download</a>

## Task 6. Copy the Source IDs for Azure AD Security Groups

The claim for the membership attribute that is sent to Cortex AgentiX uses the Object Id of the group. The Object Id is different from the Azure AD security group name. You can find the Object Id for each of your Azure AD security groups by navigating to Users and groups in Azure AD, clicking on the group name, and viewing the Object id. Create a list of the group names and corresponding Object Ids for every Azure AD security group you want to map to a Cortex AgentiX user group.

## Task 7. Configure the Cortex AgentiX SSO Integration

1. By default, SSO is disabled in Cortex AgentiX.
2. Expand the SSO Integration settings.
3. Use the following table to complete the SSO Integration settings, based on the values you saved from Azure AD.

Azure AD	Cortex AgentiX Field
Login URL	IdP SSO URL
Azure AD Identifier	IdP Issuer ID
Contents of the downloaded certificate file.	X.509 Certificate

4. In the IdP Attributes Mapping section, enter the attribute claim names from Azure AD. The names are case sensitive and must match exactly.



**NOTE:**

The attribute claim name must exactly match the value sent by your IdP. In some cases, this may be the full attribute name/namespace, depending on the configuration of our IdP

**IdP Attributes Mapping**

- Email  
[http://REDACTED/identity/claims/emailaddress](http://<b>REDACTED</b>/identity/claims/emailaddress)
- Group Membership  
[http://REDACTED/identity/claims/memberof](http://<b>REDACTED</b>/identity/claims/memberof)
- First Name  
[http://REDACTED/identity/claims/givenname](http://<b>REDACTED</b>/identity/claims/givenname)
- Last Name  
[http://REDACTED/identity/claims/surname](http://<b>REDACTED</b>/identity/claims/surname)

5. (Optional) Under Advanced Settings, select the checkboxes for ADFS and Compress encode URL (ADFS). In some circumstances, these fields may be required by your Azure AD configuration.

6. Save your settings.

**Task 8. Map SAML Group Memberships to Cortex AgentiX User Groups**

1. Right-click a user group and select Edit Group.

2. In the SAML Group Mapping field add the Azure AD group(s) Object Ids that should be associated with this user group. Multiple Object Ids should be separated with a comma. The Azure AD group Object Id must match the exact value sent in the token.

3. Save your settings.

4. Repeat for each user group.

**Task 9. Test SSO Login**

1. Go to the Cortex AgentiX tenant URL and Sign-In with SSO.

**NOTE:**

When using SAML 2.0, users are required to authenticate by logging in directly at the tenant URL. They cannot log in via Cortex Gateway.

2. After authentication to Azure AD, you are redirected again to the Cortex AgentiX tenant.

3. When logged in, validate that you have been assigned the proper roles.

To view your role and any role assigned to a user group you are a member of, click your name in the bottom left-hand corner, and click About.

## 3.6 | Engines

### Abstract

Install, manage, configure, and troubleshoot engines.

Install an engine in your remote network, enabling effortless communication with Cortex AgentiX. Easily configure and manage the engine to fit your specific needs, and explore how to leverage it for seamless integrations.

#### 3.6.1 | What is an engine?

An engine is a proxy server application that is installed on a remote machine and enables communication between the remote machine and the Cortex AgentiX tenant. You can run playbooks, scripts, commands, and integrations on the remote machine, and the results are returned to the tenant.

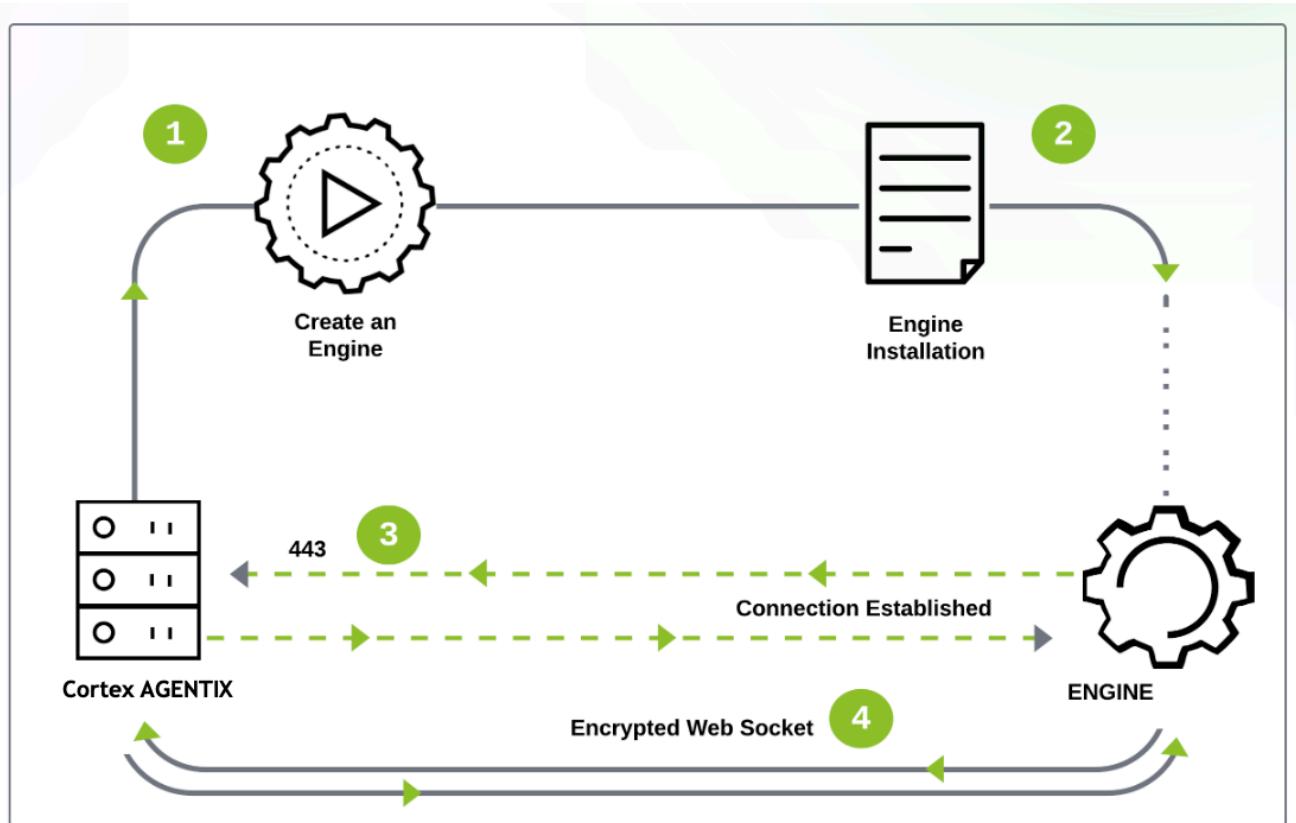
While the Cortex AgentiX tenant includes a user interface that allows security analysts to create and manage playbooks, investigate issues, and perform other tasks, the engine operates behind the scenes to execute these playbooks and automate security actions. The separation between the user interface and the engine allows for the scalable and efficient execution of security automation and orchestration.

You can install multiple engines on the same machine (Shell installation only), which is useful in a dev-prod environment where you do not want to have numerous engines in different environments and to manage those machines.

**NOTE:**

**Tip** You cannot share a multiple-engine installation with a single-engine installation.

#### Engine architecture



Within the network, you need to allow the engine to access the Cortex AgentiX's IP address and listening port (by default, TCP 443). The engine always initiates the communication to Cortex AgentiX.

#### Engine use cases

An engine can be used for the following purposes:

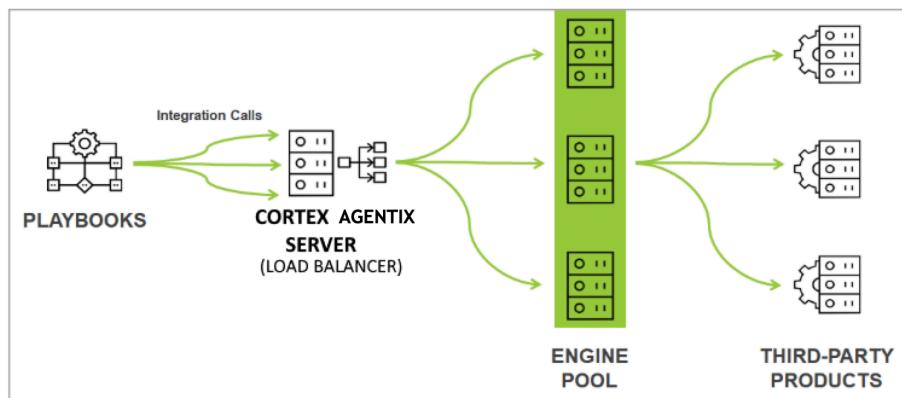


- Engine proxy

Cortex AgentIX engines enable you to access internal or external services that are otherwise blocked by a firewall or a proxy. For example, if a firewall blocks external communication and you want to run the Rasterize integration, you need to install an engine to access the Internet.

- Engine load-balancing

Engines can be part of a load-balancing group, which enables the distribution of the command execution load. The load-balancing group uses an algorithm to efficiently share the workload for integrations that the group is assigned to, thereby speeding up execution time. In general, heavy workloads are caused by playbooks that run multiple commands.



**NOTE:**

When you add an engine to a load-balancing group, you cannot use that engine separately. The engine does not appear in the engines menu when configuring an integration instance, but you can choose the load-balancing group.

### 3.6.2 | Engine requirements

#### Abstract

Hardware, OS, and required URLs for engines.

You can install engines on all Linux environments. Docker/Podman needs to be installed before installing an engine. If you are using the shell installer for an engine, Docker/Podman is installed automatically.

**NOTE:**

The Cron package is required to install engines on a Linux machine.

#### Engine hardware requirements

If your hard drive is partitioned, we recommend a minimum of 50 GB for the `/var` partition.

Component	Dev Environment Minimum	Production Minimum
CPU	8 CPU cores	16 CPU cores
Memory	16 GB RAM	32 GB RAM
Storage	100 GB	100 GB



## Operating system requirements

You can deploy a Cortex AgentiX engine on the following operating systems:

Operating System	Supported Versions
Ubuntu	18.04, 20.04, 22.04, 24.04
RHEL	8.x, 9.x Includes all minor versions.
Oracle Linux	7.x, 8.9, 9.3, 9.4
Amazon Linux	2, Amazon Linux 2023
Rocky Linux	9.5, 9.6

### NOTE:

CentOS 8.x reached End of Life (EOL) on December 31, 2021, and is no longer supported as an operating system.

CentOS 7.x reached End of Life (EOL) on June 30, 2024, and is no longer supported as an operating system.

## Engine required URLs

You need to allow the following in the URLs for Cortex AgentiX engines to operate properly. The URLs are needed to pull container images from public Docker registries.

The endpoint URL is: `wss://api-<tenant domain>.xdr.<region>.paloaltonetworks.com/xsoar/dlws`. For example,

### NOTE:

If you have configured a range of Approved IP Ranges under Allowed Sessions on the Security Settings page, the engine must communicate through one of the approved IPs.

FUNCTION	SERVICE	PORT	DIRECTION
Integrations		Integration-specific ports	Outbound
Engine connectivity	HTTPS	443 (configurable)	Outbound
Docker	<ul style="list-style-type: none"><li>• <a href="https://registry-1.docker.io">https://registry-1.docker.io</a></li><li>• <a href="https://registry.fedoraproject.org">https://registry.fedoraproject.org</a></li><li>• <a href="https://registry.access.redhat.com">https://registry.access.redhat.com</a></li><li>• <a href="https://docker.io">https://docker.io</a></li><li>• <a href="https://registry.docker.io">https://registry.docker.io</a></li><li>• <a href="https://auth.docker.io">https://auth.docker.io</a></li></ul> <p>This URL may change at Docker's discretion.</p> <ul style="list-style-type: none"><li>• <a href="https://production.cloudflare.docker.com">https://production.cloudflare.docker.com</a></li><li>• <a href="https://index.docker.io">https://index.docker.io</a></li></ul> <p>This URL may change at Docker's discretion.</p>	443	Outbound



### 3.6.3 | Install an engine

#### Abstract

Install, deploy and configure Cortex AgentiX engines.

When you install the engine, the `d1.conf` is installed on the engine machine, which contains engine properties such as proxy, log level, and log files. If Docker/Podman is already installed, the `python.engine.docker` and `powershell.engine.docker` keys are set to `true`. If Docker or Podman is not available when the engine is installed, the key is set to `false`. If so, you need to set the key to `true` after installing Docker and Podman. Verify that `python.engine.docker` and `powershell.engine.docker` configuration keys are present in the `d1.conf` file.

#### NOTE:

If you are using DEB, RPM, or Zip installation, install Docker or Podman.

Natively running Python or PowerShell integrations/scripts on Windows or Linux is not supported on Cortex AgentiX engines.

#### Installation types

Cortex AgentiX supports the following file types for installation on the engine machine:

- Shell: For all Linux deployments, including Ubuntu and SUSE. Automatically installs Docker/Podman, downloads Docker/Podman images, enables remote engine upgrade, and allows installation of multiple engines on the same machine.

The installation file is selected for you. Shell installation supports the purge flag, which by default is false. To uninstall an engine, run the installer with the purge flag enabled.

#### NOTE:

When upgrading an engine that was installed using the Shell installation, you can use the Upgrade Engine feature in the Engines page. For Amazon Linux 2 type engines, you need to upgrade these engine types using a zip-type engine and not use the Upgrade Engine feature.

If you use the shell installer, Docker/Podman is automatically installed. We recommend using Linux and not Windows to be able to use the shell installer, which installs all dependencies.

- DEB: For Ubuntu operating systems.
- RPM: RHEL operating systems.

#### NOTE:

Use DEB and RPM installation when the shell installation is not available. You need to manually install Docker or Podman and any dependencies.

- Zip: Used for Amazon Linux 2 machines.
- Configuration: Configuration file for download. When you install one of the other options, this configuration file (`d1.conf`) is installed on the engine machine.

#### IMPORTANT:

For DEB/RPM engines, Python (including 3.x) and the containerization platform (Docker/Podman) must be installed and configured. For Docker or Podman to work correctly on an engine, IPv4 forwarding must be enabled.

#### How to install an engine

1. Create an engine.
  - a. Select Settings → Configurations → Engines → Create New Engine.
  - b. In the Engine Name field, add a meaningful name for the engine.
  - c. Select one of the installer types from the list.
  - d. (Optional) (Shell only) Select the checkbox to enable multiple engines to run on the same machine.

If you have an existing engine, and you did not select the checkbox, and now you want to install another engine on the same machine, you need to delete the existing engine.

- e. (Optional) Add any required configuration in JSON format.
- f. Click OK to create the engine.

2. For shell installation, do the following:

#### TIP:

For Linux systems, we recommend using the shell installer. If using Amazon Linux 2, use the zip installer (see step 4).

- a. Move the `.sh` file to the engine machine using a tool such as SSH or PuTTY.
- b. On the engine machine, grant execution permission by running the following command:



```
chmod +x <engine-file-path>
```

c. Install the engine by typing one of the following commands:

With tools: `sudo <engine-file-path>`

Without tools: `sudo <engine-file-path> -- -tools=false`

**NOTE:**

If you receive a **permissions denied** error, it is likely that you do not have permission to access the `/tmp` directory.

If the installer fails to start due to a permissions issue, even if running as root, add one of the following two arguments when running the installer:

- `--target <path>` - Extracts the installer files into the specified custom path.

- `--keep` - Extracts the installer files into the current working directory (without cleaning at the end).

If using installer options such as `-- -tools=false`, the option should come after the `--target` or `--keep` arguments. For example:

```
sudo ./d1-installer.sh --target /some/temp/dir -- -tools=false
```

If you set a custom path when you run the installer, you must also set a custom path for upgrading your engine or the upgrade will fail. For more information, see [Upgrade an engine](#).

3. For RPM/DEB installation, do the following:

a. Move the file to the required machine using a tool such as SSH or PuTTY.

b. Type one of the following installation commands:

Machine Type	Install Command
RHEL (RPM)	<code>sudo rpm -Uvh d1-2.5_15418-1.x86_64.rpm</code>
Ubuntu (DEB)	<code>sudo dpkg --install d1_xxx_amd64.deb</code>

c. Start the engine by running one of the following commands:

Machine Type	Start Command
RHEL (RPM)	<code>sudo systemctl start d1</code>
Ubuntu (DEB)	<code>sudo service d1 restart</code>

4. For Zip installation on Amazon Linux 2, run the following commands:

a. Create the engine folder.

```
mkdir /usr/local/demisto
```

b. Unzip the engine files to the folder created in the previous step.

```
unzip ./d1.zip -d /usr/local/demisto
```

c. Allow the process to bind to low-numbered ports.

```
setcap CAP_NET_BIND_SERVICE=+eip /usr/local/demisto/d1_linux_amd64
```

d. Change the owner of `/usr/local/demisto` to the demisto user.

```
chown -R demisto:demisto /usr/local/demisto
```

e. In `/etc/systemd/system` edit the `d1.service` file as follows (adjust the directory and the name of the binary file if needed).

```
[Unit]
Description=Demisto Engine Service
After=network.target
[Service]
Type=simple
User=demisto
```



```
WorkingDirectory=/usr/local/demisto
ExecStart=/usr/local/demisto/d1_linux_amd64
EnvironmentFile=/etc/environment
Restart=always
[Install]
WantedBy=multi-user.target
```

f. Run the following commands:

```
chown root:root /etc/systemd/system/d1.service
chmod 644 /etc/systemd/system/d1.service
```

g. Run the engine process.

```
systemctl start d1
```

h. Verify that the engine is running.

```
systemctl status d1
```

5. Verify that the engine you created is connected.

a. Select Settings → Configurations → Engines.

b. Locate your engine on the Engines page and check that it is connected.

6. When the engine is connected, you can add the engine to a load-balancing group by clicking Load-Balancing Group on the Engines page.

If you want to add the engine to a new group, click Add to new group from the list.

When the engine is in the load-balancing group, it cannot be used as an individual engine and does not appear when configuring an engine from the list.

7. (Optional) After installing the engine, you may want to set up a proxy, set up Docker hardening, configure the number of workers for the engine, or perform other related engine configurations. For more information, see Configure Engines. You can also configure an integration instance to run on the engine you created.

### 3.6.3.1 | Docker

#### Abstract

Cortex AgentIX Docker installation, configuration, security, and troubleshooting guides.

Docker is a software framework for building, running, and managing containers.

#### **NOTE:**

This section is relevant when installing an engine.

Cortex AgentIX maintains a repository of Docker images, available in the Docker Hub under the Cortex organization.

Each Python/PowerShell script or integration has a specific Docker image listed in the YAML file. When the script or integration runs, if the specified Docker image is not available locally, it is downloaded from the Docker Hub or the Cortex Container Registry. The script or integration then runs inside the Docker container. For more information on Docker, see the Docker documentation and Using Docker.

#### **NOTE:**

Docker images can be downloaded together with their relevant content packs for offline installation.

### 3.6.3.1.1 | Install Docker

#### Abstract

Install Docker on engines and troubleshoot the installation.

Docker is required for engines to run Python/Powershell scripts and integrations in a controlled environment.

If you use the Shell installer to install an engine, Docker is automatically installed. If using DEB and RPM installations, install Docker or Podman before installing an engine. The engine uses Docker to run Python scripts, PowerShell scripts, and integrations in a controlled environment. By packaging libraries and dependencies together, the environment remains the same, and scripts and integrations are not affected by different server configurations.

Cortex AgentIX supports the latest Docker Engine release from Docker and the following corresponding supported Linux distributions:

- 5.3.15 and later
- 5.4.2 and later
- 5.5 and later

These Linux distributions include their own Docker Engine package. In addition, older versions of Docker Engine released within the last 12 months are supported unless there is a known compatibility issue with a specific Docker Engine version. In case of a compatibility issue, Cortex AgentIX will publish an



advisory notifying you to upgrade your Docker Engine version.

You can use a version that is not supported. However, when encountering an issue that requires Customer Support involvement, you may be asked to upgrade to a supported version before assistance can be provided.

#### Docker installation by operating system

If you need to install Docker before installing an engine, use the following procedures:

- Red Hat
- Ubuntu
- Amazon Linux
- Oracle Linux

#### NOTE:

For Red Hat's Docker distribution, you need Mirantis Container Runtime (formerly Docker Engine - Enterprise) to run specific Docker-dependent integrations and scripts. For more information, see [Install Docker distribution for Red Hat on an engine server](#).

To use the Mirantis Container Runtime (formerly Docker Engine - Enterprise) follow the deployment guide for your operating system distribution.

#### Verify Docker user and permissions

##### Verify Docker user

If you installed an engine before installing Docker, verify the `demisto` operating system user is part of the Docker operating system group.

1. Run `id demisto` verify. For example:

```
id demisto
uid=997(demisto) gid=997(demisto) groups=997(demisto),998(docker)
```

If needed, add the `demisto` user to the operating system group:

```
sudo groupadd docker
sudo usermod -aG docker demisto
```

Remove these keys from the engine configuration file.

```
python.executable
python.executable.no.docker
```

##### Verify user permissions

To verify that the operating system user (`demisto`) has the necessary permissions and can run Docker containers, run the following command from the OS command line.

```
sudo -u demisto docker run --rm -it demisto/python:1.3-alpine python --version
```

If everything is configured properly you will receive the following output. `Python 2.7.14`.

#### 3.6.3.1.1 | [Install Docker distribution for Red Hat on an engine server](#)

##### Abstract

Install Docker distribution for Red Hat.

Red Hat maintains its own package of Docker, which is the version used in OpenShift Container Platform environments, and is available in the RHEL Extras repository.

#### NOTE:

If running RHEL v8 or higher, the engine installs Podman packages and configures the operating system to enable Podman in rootless mode.

For more information about the different packages available to install on Red Hat, see the Red Hat Knowledge Base Article (requires a Red Hat subscription to access).

1. Install Red Hat's Docker package.

2. Run the following commands.

```
systemctl enable docker.service
systemctl restart docker.service
```

3. Change ownership of the Docker daemon socket so members of the `dockerroot` user group have access.

- a. Edit or create the file `/etc/docker/daemon.json`.



b. Enable OS group `dockerroot` access to Docker by adding the following entry to the `/etc/docker/daemon.json`: "group": "`dockerroot`" file. For example:

```
{ "group": "dockerroot" }
```

c. Restart the Docker service by running the following command.

```
systemctl restart docker.service
```

d. Install the engine

e. After the engine is installed, run the following command to add the `demisto` os user to the `dockerroot` os group (Red Hat uses `dockerroot` group instead of `docker`).

```
usermod -aG dockerroot demisto
```

f. Restart the engine.

#### 4. Set the required SELinux permissions.

The Cortex AgentiX engine uses the `/var/lib/demisto/temp` directory (with subdirs) to copy files and receive files from running Docker containers. By default, when SELinux is in `enforcing` mode directories under `/var/lib/` it cannot be accessed by Docker containers.

a. To allow containers access to the `/var/lib/demisto/temp` directory, you need to set the correct SELinux policy type, by typing the following command.

```
chcon -Rt svirt_sandbox_file_t /var/lib/demisto/temp
```

b. (Optional) Verify that the directory has the `container_file_t` SELinux type attached by running the following command.

```
ls -d -Z /var/lib/demisto/temp
```

c. Configure label confinement to allow Python and PowerShell containers to access other script folders.

In the `d1.conf` file, set the following parameters:

Key	Value
For Python containers	<code>python.pass.extra.keys</code>
For PowerShell containers	<code>powershell.pass.extra.keys</code>

d. Open any issue and in the issue War Room CLI, run the `/reset_containers` command.

##### 3.6.3.1.2 | Docker image security

#### Abstract

Information about Cortex AgentiX Docker image security practices.

The project that contains the source Dockerfiles used to build the images and the accompanying files is fully open source and available for review. Cortex AgentiX uses the secure Docker Hub registry for its Docker images. However, in an Engine environment, you can also use the PANW registry . You can view the Docker trust information for each image at the image info branch.

#### Docker Trust

```
Signatures for demisto/python3:3.9.1.14969
SIGNED TAG          DIGEST          SIGNERS
3.9.1.14969        b9d340c3334befc4277f4af8ceae0cbe6d8478c5fb0921d8bf98a73f525330a2  (Repo Admin)

Administrative keys for demisto/python3:3.9.1.14969
Repository Key:      dd922c0a78cf908782d68ed2f99cae9350740fc2da582a1cb5c03e530ba2dd31
Root Key:           46ab19438bffff81c925ab27466b52b175de43bad8e8a19cc878e9288246bff8d
```

We automatically update our open-source Docker images and their accompanying dependencies (OS and Python). Examples of automatic updates can be viewed on GitHub.



We maintain Docker image information, which includes information on Python packages, OS packages, and image metadata for all our Docker images. Data image information is updated nightly.

All of our images are continuously scanned using Prisma Cloud for known and newly published vulnerabilities, in two scenarios:

- Every new image, and every new version of an image, are scanned before publishing to our public registries, as part of our CI/CD process.
- All existing images are continuously scanned to check whether new vulnerabilities have been published and now exist in those images.

We evaluate all critical/high findings and actively work to prevent and mitigate security vulnerabilities.

Cortex AgentiX ensures container images are fully patched and do not contain unnecessary packages. Patches and dependencies are applied automatically via our open-source Docker file build project.

#### Response Prioritization

We remediate any critical and high level vulnerabilities, irrespective of who found them. Issues may be discovered by external researchers, found during internal testing, encountered by customers or reported by other organizations and vendors.

Any vulnerability with a possible exploitation against our images would be responded to with utmost urgency. If we conclude that there is a risk for our customers we will issue an advisory with recommended actions and mitigations. Advisories are published at: <https://security.paloaltonetworks.com/>.

In each version release (every 3 months,) we publish a new version of our content, which will use the latest and secure versions of our images.

#### Troubleshooting

- Purge old and unused images periodically.
- If you scanned the Docker images locally, and found some critical CVE's - Make sure you use the latest version of the pack, as it should have the latest version of the image. In addition, purge the old and unused images with vulnerabilities.

#### 3.6.3.1.1.3 | Docker FAQs

##### Abstract

Frequently asked questions (FAQ) about Docker in Cortex AgentiX.

- **Does Cortex AgentiX use COPY or ADD for building images?**

Cortex AgentiX uses COPY for building images. The COPY instruction copies files from the local host machine to the container file system. Cortex AgentiX does not use the ADD instruction, which could potentially retrieve files from remote URLs and perform operations such as unpacking, introducing potential security vulnerabilities.

- **Should the --restart flag be used?**

The --restart flag should not be used. Cortex AgentiX manages the lifecycle of Docker images and restarts images as needed.

- **Can we restrict containers from acquiring additional privileges by setting the no-new-privileges option?**

Cortex AgentiX does not support the no-new-privileges option. Some integrations and scripts may need to change privileges when running as a non-root user (such as Ping).

- **Can we apply a daemon-wide custom seccomp profile?**

The default seccomp profile from Docker is strongly recommended. The default seccomp profile provides protection as well as wide application compatibility. While you can apply a custom seccomp profile, Cortex AgentiX cannot guarantee that it won't block system calls used by an integration or script. If you apply a custom seccomp profile, you need to verify and test the profile with any integrations or scripts you plan to use.

- **Can we use TLS authentication for docker daemon configuration?**

TLS authentication is not used, because Cortex AgentiX does not use Docker remote connections. All communication is done via the local Docker IPC socket.

- **How do we set the logging level to info?**

Set the log level in the Docker daemon configuration file.

- **Can we restrict Linux kernel capabilities within containers?**

The default Docker settings (recommended) include 14 kernel capabilities and exclude 23 kernel capabilities. Refer to Docker's full list of runtime privileges and Linux capabilities.

You can further exclude capabilities via advanced configuration, but will first need to verify that you are not using a script that requires the capability. For example, Ping requires `NET_RAW` capability.



- Is the Docker health check option implemented at runtime?

The Cortex AgentIX tenant monitors the health of the containers and restarts/terminates containers as needed. The Docker health check option is not needed.

- Can we enable live restore?

Live restore is not used. Cortex AgentIX uses ephemeral Docker containers. Every running container is stateless by design.

- Can we restrict network traffic between containers?

Cortex AgentIX does not disable inter-container communication by default, as there are use cases where this might be needed. For example, a script communicating with a long running integration which listens on a port, may require inter-container communication. If inter-container communication is not required, it can be disabled by modifying the Docker daemon configuration.

- Can we enable user namespace remapping?

Cortex AgentIX does not support user namespace remapping.

- How do we configure auditing for Docker files and directories?

Auditing is an operating system configuration, and can be enabled in the operating system settings. Cortex AgentIX does not change the audit settings of the operating system.

- Can we disable the userland proxy?

If the kernel supports hairpin NAT, you can disable docker userland proxy settings by modifying the Docker daemon configuration.

- Does Cortex AgentIX support the AppArmor profile?

Cortex AgentIX supports the default AppArmor profile (only relevant for Ubuntu with AppArmor enabled).

- Does Cortex AgentIX support the SELinux profile?

Cortex AgentIX supports the default SELinux profile (only relevant for RedHat with SELinux enabled).

- How does Cortex AgentIX handle secrets management?

For Docker swarm services, a secret is a blob of data, such as password, SSH private keys, SSL certificates, or other piece of data that should not be transmitted over a network or stored unencrypted in a Docker file or in your application's source code. Cortex AgentIX manages integration credentials internally. It also supports using an external credentials service such as CyberArk.

#### 3.6.3.1.1.4 | Troubleshoot Docker issues

The following provides troubleshooting solutions for Docker networking and performance issues.

##### Troubleshoot Docker networking issues

In Cortex AgentIX, integrations and scripts run either on the tenant, or on an engine.

If you have Docker networking issues when using an engine, you need to modify the `d1.conf` file.

1. On the machine where the Engine is installed, open the `d1.conf` file.

2. Add the following to the `d1.conf` file:

```
{
  "LogLevel": "info",
  "LogFile": "/var/log/demisto/d1.log",
  "EngineURLs": [
    "wss://1234.demisto.live/dlws"
  ],
  "BindAddress": ":443",
  "EngineID": "XYZ",
  "ServerPublic": "ABC",
  "ArtifactsFolder": "",
  "TempFolder": "",
  "python.pass.extra.keys": "--network=host"
}
```

3. Save the file.

4. Restart the engine using `systemctl restart d1` or `service d1 restart`.

##### Troubleshoot Docker performance issues

This information is intended to help resolve the following Docker performance issues.



- Containers are getting stuck.
- The Docker process consumes a lot of resources.
- Time synchronization issues between the container and the operating system.

#### Cause

The installed Docker package and its dependencies are not up to date.

#### Workaround

1. Update the package manager cache.

Linux Distribution	Command
Debian	<code>apt-get update</code>

2. (Optional) Check for a newer version of the Docker package.

Linux Distribution	Command
Debian	<code>apt-cache policy docker</code>

3. Update the Docker package.

Linux Distribution	Command
Debian	<code>apt-get update docker</code>

#### 3.6.3.1.5 | Configure Docker pull rate limit

#### Abstract

Configure the Docker pull rate limit on public images. Create a Docker user account and receive a higher pull limit.

Docker enforces a pull rate limit on public images. The limit is based on an IP address or as a logged-in Docker hub user. The default limit (100 pulls per 6 hours) is usually high enough for Cortex AgentX's use of Docker images, but the rate limit may be reached if using a single IP address for a large organization (behind a NAT). If the rate limit is reached, the following error message is issued:

```
Error response from daemon: toomanyrequests: You have reached your pull rate limit. You may increase the limit by authenticating and upgrading: https://www.docker.com/increase-rate-limit.
```

To increase the limit:

1. Sign up a free user in the Docker hub.

The pull limit is higher for a registered user (200 pulls per 6 hours).

2. Authenticate the user on the engine machine by running the following command.

```
sudo -u demisto docker login
```

3. (Optional) Instead of manually logging in to Docker to pull images, you can edit the Docker config file to use credentials from the file or from a credential store.

#### 3.6.3.1.6 | Change the Docker installation folder

#### Abstract

Instructions for changing the default Docker folder.



The `/var/lib/docker/` folder is the default Docker folder for Ubuntu, Fedora, and Debian in a standard engine installation.

To change the Docker folder:

1. Stop the Docker daemon.

```
sudo service docker stop
```

2. Create a file called `daemon.json` under the `/etc/docker` directory with the following content:

```
{  
    "data-root": "<path to your Docker folder>"  
}
```

3. Copy the current data directory to the new one.

```
sudo rsync -ap /var/lib/docker/ <path to your Docker folder>
```

4. Rename the old docker directory.

```
sudo mv /var/lib/docker /var/lib/docker.bkp
```

5. After confirming that the change was successful, you can remove the backup file.

```
sudo rm -rf /var/lib/docker.bkp
```

6. Start the Docker daemon.

```
sudo service docker start
```

#### 3.6.3.1.2 | Docker hardening guide

##### Abstract

Use the Docker Hardening Guide to configure the Cortex AgentX settings when running Docker containers.

The following describes the engine settings we recommend for securely running Docker containers.

When editing the configuration file, you can limit container resources, open file descriptors, limit available CPU, and more. For example, add the following keys to the configuration file:

```
{"docker.run.internal.asuser": true, "limit.docker.cpu": true, "limit.docker.memory": true, "python.pass.extra.keys": "--pids-limit=256##--ulimit=nofile=1024:8192"}
```

##### TIP:

We recommend reviewing *Docker network hardening* below before changing any parameters in the configuration file.

To securely run Docker containers, we recommend using the latest Docker version.

You can *Check Docker Hardening Configurations* to verify that the Docker container has been hardened according to the settings we recommend.

##### NOTE:

The settings below can also be applied to Podman, with the exception of limiting available memory, limiting available CPU, and limiting PIDS.

##### Docker network hardening

Docker creates its networking stack that enables containers to communicate with other networking endpoints. You can use iptables rules to restrict which networking sources the containers communicate with. By default, Docker uses a networking configuration that allows unrestricted communication for containers, so that containers can communicate with all IP addresses.

- Block network access to the host machine

Integrations and scripts running within containers do not usually require access to the host network. For added security, you can block network access from containers to services running on the engine machine.

For example, to limit all source IPs from containers that use the IP ranges 172.16.0.0/12, run `sudo iptables -I INPUT -s 172.16.0.0/12 -d 10.18.18.246 -j DROP`. This also ensures that new Docker networks that use addresses in the IP address range of 172.16.0.0/12 are blocked from access to the host private IP. The default IP range used by Docker is 172.16.0.0/12. If you configured a different range in Docker's `daemon.json` config file, use the configured range. Alternatively, you can limit specific interfaces by using the interface name, such as `docker0`, as a source.

1. Add the following iptables rule for each private IP on the tenant machine:

```
sudo iptables -I INPUT -s <IP address range> -d <host private ip address> -j DROP
```

2. (Optional) To view a list of all private IP addresses on the host machine, run `sudo ifconfig -a`



- Assign a Docker network for a Docker image

If your engine is installed on a cloud provider such as AWS or GCP, it is best practice to block containers from accessing the cloud provider's instance metadata service. The metadata service is accessed via IP address 169.254.169.254. For more information about the metadata service and the data exposed, see the AWS and GCP documentation

There are cases where you might need to provide access to the metadata service. For example, access is required when using an AWS integration that authenticates via the available role from the instance metadata service. You can create a separate Docker network, without the blocked iptable rule, to be used by the AWS integration's Docker container. For most AWS integrations, the relevant Docker image is: demisto/boto3py3

1. Create a new Docker network by running the following command:

```
sudo docker network create -d bridge -o com.docker.network.bridge.name=docker-metadata aws-metadata
```

2. Edit the engine configuration file either by editing the d1.conf file, or If you installed via Shell, you can edit the configuration in the UI as well as editing the file directly. For details, see Configure engines.

3. Add the following key.

```
"python.pass.extra.keys.demisto/boto3py3": "--network=aws-metadata"
```

4. Save the changes.

5. Restart the demisto service on the engine machine.

```
sudo systemctl start d1
```

```
(Ubuntu) sudo service d1 restart
```

6. Verify the configuration of your new Docker network:

```
sudo docker network inspect aws-metadata
```

- Block internal network access

In some cases, you might need to block specific integrations from accessing internal network resources and allow the integrations to access only external IP addresses. We recommend this setting for the Rasterize integration when used to Rasterize untrusted URLs or HTML content, such as those obtained via external emails. With internal network access blocked, a rendered page in the Rasterize integration cannot perform an SSRF or DNS rebind attack to access internal network resources.

1. Create a new Docker network by running the following command:

```
sudo docker network create -d bridge -o com.docker.network.bridge.name=docker-external external
```

2. Block network access to the host machine for the new Docker network:

```
iptables -I INPUT -i docker-external -d <host private ip> -j DROP
```

3. Block network access to cloud provider instance metadata:

```
sudo iptables -I DOCKER-USER -i docker-external -d 169.254.169.254/32 -j DROP
```

4. Block internal network access:

```
sudo iptables -I DOCKER-USER -i docker-external -d 10.0.0.0/8 -j DROP
```

```
sudo iptables -I DOCKER-USER -i docker-external -d 172.16.0.0/12 -j DROP
```

```
sudo iptables -I DOCKER-USER -i docker-external -d 192.168.0.0/16 -j DROP
```

5. Edit the engine configuration file either by editing the d1.conf file, or If you installed via Shell, you can edit the configuration in the UI as well as editing the file directly. For details, see Configure engines.

6. Add the following key to run integrations that use the demisto/chromium Docker image with the Docker network external.

```
"python.pass.extra.keys.demisto/chromium": "--network=external"
```

7. Save the changes.

8. Restart the demisto service on the engine machine.

```
sudo systemctl start d1
```

```
(Ubuntu) sudo service d1 restart
```

9. Verify the configuration of your new Docker network:

```
sudo docker network inspect external
```



- **Persist iptables rules**

By default, iptables rules are not persistent after a reboot. To ensure your changes are persistent, save the iptables rules by following the recommended configuration for your Linux operating system:

- Ubuntu
- Red Hat and related operating system flavors

#### Configure Docker images

You can apply more specific fine tuned settings to Docker images, according to the Docker image name or the Docker image name including the image tag. To apply settings to a Docker image name, add the advanced configuration key to the engine configuration file. If you apply Docker image specific settings, they will be used instead of the general `python.pass.extra.keys` setting. This overrides the general memory and CPU settings, as needed.

1. Edit the engine configuration file either by editing the `d1.conf` file, or if you installed via Shell, you can edit the configuration in the UI as well as editing the file directly. For details, see [Configure engines](#).
2. Add the following key to apply settings to a Docker image name.

```
"python.pass.extra.keys.<image_name>"
```

For example, `"python.pass.extra.keys.demisto/dl"`.

- To apply settings to a Docker image name, including the image tag, use `"python.pass.extra.keys.<image_name>": "<image_tag>"`.

For example, `"python.pass.extra.keys.demisto/dl": "1.4"`.

- To set the Docker images `demisto/dl` (all tags) to use a higher max memory value of 2g and to remain with the recommended PIDs and ulimit, add the following to the configuration file:`"python.pass.extra.keys.demisto/dl": "--memory=2g##--ulimit=no-file=1024:8192##--pids-limit=256"`

3. Save the changes.

4. Restart the demisto service on the engine machine.

```
sudo systemctl start d1
```

```
(Ubuntu) sudo service d1 restart
```

#### Run Docker with non-root internal users

For additional security isolation, we recommend to run Docker containers as non-root internal users. This follows the principle of least privilege.

1. Edit the engine configuration file either by editing the `d1.conf` file, or if you installed via Shell, you can edit the configuration in the UI as well as editing the file directly. For details, see [Configure engines](#).

2. Add the following key:

```
"docker.run.internal.asuser": true
```

3. For containers that do not support non-root internal users, add the following key:

```
"docker.run.internal.asuser.ignore" : "A comma separated list of container names. The engine matches the container names according to the prefixes of the key values"
```

For example, `"docker.run.internal.asuser.ignore"="demisto/python3:","demisto/python:"`

The engine matches the key values for the following containers:

```
demisto/python:1.3-alpine
demisto/python:2.7.16.373
demisto/python3:3.7.3.928
demisto/python3:3.7.4.977
```

The `:` character should be used to limit the match to the full name of the container. For example, using the `:` character does not find `demisto/python-ubuntu:2.7.16.373`.

4. Save the changes.

5. Restart the demisto service on the engine machine.

```
sudo systemctl start d1
```

```
(Ubuntu) sudo service d1 restart
```

#### Configure the memory limit support without swap limit capabilities

When a container exceeds the specified amount of memory, the container starts to swap. Not all Linux distributions have the swap limit support enabled by default.



- Red Hat distributions usually have swap limit support enabled by default.
- Ubuntu distributions usually have swap limit support disabled by default.

To protect the host from a container using too many system resources (either because of a software bug or a DoS attack), limit the resources available for each container. In the engine configuration file, some of these settings are set using the advanced parameter: `python.pass.extra.keys`. This key receives as a parameter full `docker run` options, separated with the `##` string.

How to check if your system supports swap limit capabilities

1. On the engine machine, run the following command:

```
sudo docker run --rm -it --memory=1g demisto/python:1.3-alpine true
```

2. If `swap limit capabilities` is enabled, configure the memory limitation . (To test the memory, see step 5 of configure the memory limitation.)

3. If you see the following message in the output (the message may vary between Docker versions):

```
WARNING: Your kernel does not support swap limit capabilities or the cgroup is not mounted. Memory limited without swap.
```

You have 2 options:

- Configure `swap limit capabilities` by following the Docker documentation.
- See How to configure the memory limit support without swap limit capabilities.

If you see the `WARNING: No swap limit support` you can configure memory support without swap limit capabilities.

How to configure the memory limit support without swap limit capabilities

1. Edit the engine configuration file either by editing the `d1.conf` file, or If you installed via Shell, you can edit the configuration in the UI as well as editing the file directly. For details, see Configure engines.

2. Add the following key to disable swap memory enforcement:

```
"python.pass.extra.keys": "--memory=1g##--memory-swap=-1"
```

If you have the `python.pass.extra.keys` already set up with a value, add the value after the `##` separator.

3. Save the changes.

4. Restart the demisto service on the engine machine.

```
sudo systemctl start d1
```

```
(Ubuntu) sudo service d1 restart
```

Configure the memory limitation

We recommend limiting available memory for each container to 1 GB.

If `swap limit capabilities` is enabled (see How to check if your system supports swap limit capabilities above), in Cortex AgentX configure the memory limitation using the following advanced parameters.

1. Edit the engine configuration file either by editing the `d1.conf` file, or If you installed via Shell, you can edit the configuration in the UI as well as editing the file directly. For details, see Configure engines.

2. Add the following keys.

```
"limit.docker.memory": true, "docker.memory.limit": "1g"
```

#### **NOTE:**

If you do not want to apply Docker memory limitations, you should explicitly set the advanced parameter: `limit.docker.memory` to `false`.

3. Save the changes.

4. Restart the demisto service on the engine machine.

```
sudo systemctl start d1
```

```
(Ubuntu) sudo service d1 restart
```

5. Test the memory limit.

1. In the Script Name file, type `TestMemory`.

2. Add the following script:



```

from multiprocessing import Process
import os

def big_string(size):
    sys.stdin = os.fdopen(0, "r")
    s = 'a' * 1024
    while len(s) < size:
        s = s * 2
    print('completed creating string of length: {}'.format(len(s)))

size = 1 * 1024 * 1024 * 1024
p = Process(target=big_string, args=(size, ))
p.start()
p.join()
if p.exitcode != 0:
    return_error("Return code from sub process indicates failure: {}".format(p.exitcode))
else:
    print("Success allocating memory of size: {}".format(size))

```

3. In the SCRIPT SETTINGS section, select the script to run on the Single engine and select the engine where you want to run the script.

4. Save the script.

5. To test the memory limit, type `!TestMemory`. The command returns an error when it fails to allocate 1 GB of memory.

Configure the CPU, PIDs, and open file descriptors limit

Set the advanced parameters to configure the CPU limit, PIDs limit, and the open file descriptor limit.

1. Edit the engine configuration file either by editing the `d1.conf` file, or if you installed via Shell, you can edit the configuration in the UI as well as editing the file directly. For details, see Configure engines.

2. Add the following keys:

Parameter	Key
Available CPU limit	<code>"limit.docker.cpu": true, "docker.cpu.limit": "&lt;CPU Limit&gt;"</code> We recommend to limit each container to 1 CPU. (For example, <code>1.0</code> . Default is <code>1.0</code> ).
PIDs limit	<code>"python.pass.extra.keys": "--pids-limit=256"</code>
Open file descriptors limit	<code>"python.pass.extra.keys": "--ulimit=nofile=1024:8192"</code>

3. Save the changes.

4. Restart the demisto service on the engine machine.

```

sudo systemctl start d1
(Ubuntu) sudo service d1 restart

```

Check Docker hardening configurations

Check your Docker hardening configurations on an engine by running the `!DockerHardeningCheck` command in the Case/Issue War Room CLI. The results show the following:

- Non-root User
- Memory
- File descriptors
- CPUs
- PIDs

Before running the command, ensure that your engine is up and running.

1. Update the `DockerHardeningCheck` script to run on the engine.

- Go to Investigation & Response → Automation → Scripts → DockerHardeningCheck → Settings.
- In the Run on field select Single engine and from the list, select the engine you want to run the script.



- c. Save the script.
2. Verify the Docker container has been hardened according to recommended settings. In the Case/Issue War Room CLI, run the `!DockerHardeningCheck` command.

### 3.6.3.2 | Podman

#### Abstract

Run Podman containers instead of Docker for RHEL v8.

Podman is a daemonless container engine for developing, managing, and running OCI Containers on Linux systems. Containers can either be run as root or in rootless mode.

If you use the Shell installer to install an engine, Cortex AgentX automatically detects the container management type based on the operating system. For example, if your operating system is running RHEL v8 and higher, Cortex AgentX installs Podman packages and configures the operating system to enable Podman in rootless mode.

#### NOTE:

When upgrading an engine, the engine keeps the previously used container management type (regardless of distribution version).

If using PowerShell integrations, you may need to configure the default SELinux policy as Podman can affect processes that `mmap` to `/dev/zero`.

#### Docker hardening guidelines

Docker hardening guidelines can be applied to Podman, except Limit Available Memory, Limit Available CPU, and Limit PIDS.

### 3.6.3.2.1 | Change the container storage directory

By default, Podman uses the `$HOME/.local/share/containers/storage` directory. To use a different directory for container storage, edit the Podman config file located at `/home/demisto/.config/containers/storage.conf`. If the Podman config file does not exist, you need to create it and change the ownership.

The new storage directory needs to be owned by the `demisto` user, otherwise, they will be denied access to it.

#### WARNING:

Do not use NAS storage or a temporary (`tmpfs`) directory for the `graphroot` setting. The `graphroot` needs to be a local, non-temporary directory for Podman to work. For more information, see [https://en.wikipedia.org/wiki/Network-attached\\_storage](https://en.wikipedia.org/wiki/Network-attached_storage).

#### TIP:

We recommend reserving 150 GB for container storage, either in the `/home` partition or a different storage directory that you have set using the `graphroot` key.

1. If the Podman config file does not exist:

- a. Create the Podman config file.

```
sudo mkdir -p /home/demisto/.config/containers  
cp /etc/containers/storage.conf /home/demisto/.config/containers
```

- b. Change the ownership of the Podman config file.

```
sudo chown -R demisto:demisto /home/demisto
```

2. To set a different directory for container storage, change the key: `graphroot` in the `storage.conf` file. For example:

```
graphroot = "/var/lib/containers/cortex-storage"
```

3. Some additional changes are required in the `storage.conf` file. Comment out the `runroot` setting by adding a # (hash) before it. For example:

```
#runroot = "/run/containers/storage"
```

#### NOTE:

Alternatively, the `runroot` setting may be set to some temporary directory that is accessible by the user `demisto`. If you choose to set the `runroot`, it must be a directory that is mounted as `tmpfs` (temporary filesystem), unlike the `graphroot`.

4. Under `[storage.options.overlay]`, uncomment the following line (remove the # from the start):

```
mount_program = "/usr/bin/fuse-overlayfs"
```

5. If the engine has already been installed, apply your changes to any existing containers:

```
sudo -u demisto podman system migrate
```



6. Verify the change (once the engine is installed):

```
sudo -u demisto podman info | grep graph
```

#### 3.6.3.2.2 | Install Podman

##### Abstract

Install Podman on engines for RHEL v8 or later.

When installing a new engine on RHEL 8 or later, the shell installer configures Podman automatically. There are some cases, however, where you might need to install Podman manually:

- When using an installation method other than the shell installer (e.g., an RPM package) on RHEL 8 or later.
- When the shell installer did not successfully install Podman.
- When you want to migrate from Docker to Podman for an existing Cortex AgentX engine.

##### NOTE:

- This procedure is intended for RHEL 8 or later. It may not work for other operating system types.
- Do not use NAS storage for the \$HOME directory. The directory needs to be a local directory for Podman to work.

1. For RHEL 8, install Podman by typing the following commands:

- sudo yum -y install slirp4netns fuse-overlayfs
- sudo yum -y module install container-tools

For RHEL 9 or later, install Podman by typing the following command:

- sudo yum -y install slirp4netns fuse-overlayfs podman

2. Run the following commands:

- sudo touch /etc/subuid /etc/subgid
- sudo mkdir -p /home/demisto
- sudo chown demisto:demisto /home/demisto

3. Configure the `unqualified-search-registries` used by Podman.

Podman by default uses the fedoraproject.org, redhat.com, and docker.io unqualified search registries. Since Cortex AgentX images use only the docker.io registry, you can speed up download times for container images by setting `unqualified-search-registries` to just docker.io.

a. Create or edit the `/home/demisto/.config/containers/registries.conf` config file.

b. In the file, set `unqualified-search-registries = ["docker.io"]`

##### NOTE:

If you edit the file with the `root` user, make sure to set the `demisto` user as file owner by running `chown demisto:demisto /home/demisto/.config/containers/registries.conf`

4. Change the `subuids` and `subgids` by running the following command:

```
sudo usermod --add-subuids 200000-265535 --add-subgids 200000-265535 demisto
```

5. Migrate existing containers to Podman:

```
sudo sh -c "cd /; runuser -u demisto -- podman system migrate"
```

6. Set the `net.ipv4.ping_group_range`, by typing the following commands:

- sudo sh -c "echo 'net.ipv4.ping\_group\_range=0 2000000' > /etc/sysctl.d/demisto-ping.conf"
- sudo sysctl -w "net.ipv4.ping\_group\_range=0 2000000"

7. As root user, edit the following `config` file:

```
/usr/local/demisto/d1.conf
```

8. Change the "`container.engine.type": "docker"` to `"podman"`.

If this line does not exist, add the following line to the file:



```

"container.engine.type": "podman"

"Server": {
    "HttpsPort": "443",
    "ProxyMode": true
},
"container": {
    "engine": {
        "type": "podman"
    }
},
"db": {
    "index": {
        "entry": {
            "disable": true
        }
    }
}
}

```

9. If the engine is running, restart the service.

```
sudo systemctl restart d1
```

#### **NOTE:**

If the Allow running multiple engines on the same machine option is selected, run the command:

```
sudo systemctl restart d1_<Engine _name>
```

#### 3.6.3.2.3 | Migrate From Docker to Podman

##### Abstract

Switch from Docker to Podman when installing an engine for RHEL 8 or later.

Although Podman is set up automatically in an engine installation, it is possible to migrate from Docker to Podman in an existing engine. Follow the Podman installation instructions to migrate.

#### 3.6.3.2.4 | Troubleshoot Podman

##### Abstract

Troubleshoot process leaks or installation issues for Podman in Cortex AgentiX,

dbus-daemon process leak

Podman version 3.4.1 and lower has a known issue that dbus-daemon processes may leak when running in an environment containing the dbus-x11 OS package. The issue occurs when the dbus-x11 OS package is installed, for example, when installing an X11 desktop environment like GNOME desktop on the host machine. If you experience this issue, you see a large number of dbus-daemon processes owned by the demisto OS user. To check if you are affected by the issue, run the following command:

```
ps -fe | grep demisto | grep dbus-daemon
```

To fix this issue:

1. Remove the dbus-x11 OS package and dependent packages by running the following command:

```
sudo yum remove dbus-x11
```

2. After removal you can kill the leaked dbus-daemon processes by running the following OS command:

```
pgrep -u demisto dbus-daemon | xargs sudo kill
```

Invalid argument error

When Podman fails to run with an “ Invalid argument” error, such as:

```
ERRO[0000] running `/usr/bin/newuidmap 15936 0 1029 1 1 165536 65536 65537 200000 65536`: newuidmap: write to uid_map failed: Invalid argument
Error: cannot set up namespace using "/usr/bin/newuidmap": exit status 1
```

This can be caused by duplicate lines for Cortex AgentiX in /etc/subuid and /etc/subgid.

To fix this issue:

1. Check if the /etc/subuid file contains multiple lines that start with the Cortex AgentiX username (usually demisto). For example:

```
alice:100000:65536
demisto:165536:65536
demisto:200000:65536
splunk:331072:65536
```



2. If this is the case, edit the file as root, and remove the extra line(s) for Cortex AgentiX. The line you should keep is the one that ends with 200000:65536.  
Continuing with the above example, here is the end result:

```
alice:100000:65536
demisto:200000:65536
splunk:331072:65536
```

3. Repeat the above steps for the /etc/subgid file.

Verify Podman installation

When encountering errors in Cortex AgentiX that are Podman related, such as:

- failed to run "docker ps". stderr: [], err: [Timeout. Process killed (1400)]
- Timeout while waiting for pong response [error 'Read timed out (15s)']
- Error: error joining network namespace of container  
06b8aec6eabe2e735128e3a72cb06c8ae2d97ade60a56ab555034442ea4e2a84: error retrieving network namespace at  
/tmp/podman-run-989/netns/cni-86dca01c-bd84-1aaf-85fb-72b659a8e42a: unknown FS magic on "/tmp/podman-run-993/netns/cni-86dca01c-bd84-1aaf-85fb-72b659a8e42a": 58465342

1. Verify that Podman is running properly with the `demisto` OS user by performing the following steps:

- Change the OS user to `demisto` by running the following command:

```
sudo su - -s /bin/bash demisto
```

- Check that your system complies with the minimum requirements, and view general system information such as host architecture, CPU, OS, registries, container storage path, etc., by running the following command:

```
podman info
```

- Check all active running containers, container names, and IDs by running the following command:

```
podman ps
```

- Check that Podman can run a container by running the following command:

```
podman run --rm -t demisto/python3:3.10.4.29342 echo "podman is working"
```

If any of the Podman commands are not working, try running with the `--log-level=debug` to receive additional details as to why it is failing. For example:

```
podman --log-level=debug ps
```

```
podman --log-level=debug ps podman --log-level=debug run --rm -t demisto/python3:3.10.4.29342 echo "podman is working"
```

2. Reset the Podman Data Directories.

If the Podman commands in step 1 are failing, you should clean the Podman working directories. Sometimes Podman's data directories get corrupted (for example, as a result of insufficient disk space).

**NOTE:**

This step removes all Podman images, including any custom images you may have created.

- a. Stop the engine by running the following command:

```
sudo systemctl stop d1
```

- b. Ensure that all Podman containers of the `demisto` user are stopped by running the following command:

```
ps -fe | grep demisto | grep 'podman run'
```

If required, kill the running containers.

- c. Delete the following directories (assuming the `demisto` OS user's home directory is at: /home/demisto)

- sudo rm -rf /home/demisto/.cache/containers/
- sudo rm -rf /home/demisto/.local/share/containers/
- sudo rm -rf /tmp/podman-run-\$(id -u demisto)
- sudo rm -rf /tmp/containers-user-\$(id -u demisto)
- sudo rm -rf /tmp/tmp/run-\$(id -u demisto)



**NOTE:**

`$(id -u demisto)` is used to get the `demisto` user ID, which is part of the directory name. For example, `/tmp/podman-run-993`

Not all the directories above may be present.

d. Start the engine by running the following command:

```
sudo systemctl start d1
```

e. Verify that Podman is working properly with the `demisto` OS user by following step 1.

Unused containers are taking up resources

In some cases, if the Podman process crashes or is killed abruptly, it can leave containers on disk. You might see errors such as `error allocating lock for new container: allocation failed; exceeded num_lock` when the maximum number of locks used to manage containers is exhausted due to the unused containers that remain.

1. Change to the demisto operating system user `sudo su - -s /bin/bash demisto`.

2. Run `podman ps -a -f status=exited` to check for unused containers.

3. Clean up the unused containers `podman container cleanup --rm -a`.

**NOTE:**

When you run `podman container cleanup --rm -a`, you might see a message such as `running or paused containers cannot be moved without force`. The message can be safely ignored, as it only pertains to current running containers, which are not removed.

4. After cleanup, verify there are no remaining unused containers `podman ps -a -f status=exited`.

Keyring quota exceeded error

```
Script failed to run: Docker code runner got container error: [Docker code script is in inconsistent state, ... error: [exit status 126] stderr: [Error: OCI runtime error: crun: create keyring .... Disk quota exceeded]
```

By default, Podman creates a `keyring` that is used by each container. The limit per user on the machine might be low, and Podman can reach the limit when running more containers than the `keyring` limit. To check the `keyring` usage, run the `sudo cat /proc/key-users` operating system command.

The command returns the usage for each UID (to retrieve the demisto user UID, run `id demisto`). The fourth column shows the number of keys used out of the total number available. For more information about keys, see Kernel Key Retention Service.

You can either increase the limit of max keyrings (increasing to 1000 is safe and reasonable) per user, as specified by your Linux vendor documentation or you can disable keyring creation by Podman. We recommend disabling keyring creation unless keyrings are used by Podman in other applications on the machine. To disable keyring creation by Podman, modify the `containers.conf` file and add the option `keyring = false` under the "[containers]" section. For more information, see the Containers Engine Configuration File.

error "exit status 125" and output "Error: chown ... operation not permitted"

If the container storage directory is not owned AND exclusively used by the demisto user, scripts will fail to run. See the Podman section for more information about assigning ownership of the storage directory.

Report a support case for installation issues

If the procedure set out in the Verify Podman installation section above does not solve the Podman issue and you require assistance from Support, do the following:

1. Include the following files as part of the support case:

- `/etc/containers/storage.conf`
- `/home/demisto/.config/containers/storage.conf`

If the file does not exist, indicate that there is no such file.

- `/home/demisto/.config/containers/registries.conf`

If the file does not exist, indicate that there is no such file.

2. Include the output of the following commands as the `demisto` user.

**NOTE:**

To change to the `demisto` OS user, run the following command:

```
sudo su - -s /bin/bash demisto
```



- `podman info`
- `podman images`
- `podman --log-level=debug ps`
- `podman --log-level=debug run --rm -t demisto/python3:3.10.4.29342 echo "podman is working"`

Permission issues with directories under the `/run` path

When installing a Cortex AgentIX engine on a RHEL system (version 8 or later), or when running an integration on such an engine, you get a permission error for a path under `/run` (for example `/run/user/0` or `/run/libpod`).

1. In RHEL 9 only: Make sure the `container-tools` meta-package is installed by running:

```
yum -y install container-tools
```

2. Run the following commands:

- `cp /etc/containers/storage.conf /home/demisto/.config/containers/storage.conf`
- `chown demisto:demisto /home/demisto/.config/containers/storage.conf`
- `chmod 600 /home/demisto/.config/containers/storage.conf`

3. Edit `/home/demisto/.config/containers/storage.conf`.

- Under `[storage]`, change `runroot` to some temporary directory that is accessible by user `demisto`.

For example: `runroot = "/tmp/podman-run-agentix"`

#### **IMPORTANT:**

The `runroot` must be located under the `tmpfs` file system type. This is required to clean Podman's run state on reboot and for performance reasons.

- Also under `[storage]`, change `graphroot` (where container images are stored) to any location that is owned and accessible by user `demisto`. We recommend using this standard path:

```
graphroot = "/home/demisto/.local/share/containers/storage"
```

#### **CAUTION:**

Unlike the `runroot`, the `graphroot` must NOT be located under the `tmpfs` file system type. Using `tmpfs` for the `graphroot` might corrupt container images, causing command executions to fail. It also degrades performance by forcing Podman to needlessly re-pull images.

- Under `[storage.options.overlay]`, uncomment the following line (remove the `#` from the start):

```
mount_program = "/usr/bin/fuse-overlayfs"
```

4. Save the file and run the following.

#### **NOTE:**

You must switch to user `demisto` before running the "system migrate" (running it as root will have no effect).

- `su - demisto`
- `podman system migrate`

5. Also as user `demisto`, run the following to ensure the path changes were applied:

```
podman info | grep Root
```

You should see the correct `runRoot` and `graphRoot` settings.

6. As user `demisto`, verify the issue is resolved by running:

```
podman run hello-world
```

7. If the issue persists, purge Podman's database by running the following:

#### **NOTE:**

The "system migrate" must be done by the user `demisto`.

- `rm -rf /home/demisto/.local/share/containers/*`
- `podman system migrate`



### 3.6.4 | Manage engines

#### Abstract

Manage engines and load balancing groups.

You can manage your engines and load-balancing groups by going to Settings → Configurations → Engines.

You can view engine names, hosts, status, connection, and other engine information.

You can do the following:

Option	Description
Load-Balancing Group	<p>It is useful to create separate load-balancing groups. For example:</p> <ul style="list-style-type: none"><li>• Use separate load-balancing groups for different integrations and instances. Create Load-Balancing groups for certain tasks, which can help segregate the infrastructure of critical integrations.</li><li>• Managed Security Service Providers may want to split internal engines and SaaS product engines.</li><li>• If you have multiple AWS accounts that are not connected and do not want a single point of failure for AWS integrations that use STS.</li></ul> <p>You can do the following:</p> <ul style="list-style-type: none"><li>• Add/remove engines to a load-balancing group</li></ul> <p>You can only add the engine to the load-balancing group after you have connected the engine.</p> <p>If you want to remove the last engine from a specific load-balancing group, if one or more integration instances use that engine, you will get an error. Before moving the engine, you need to assign Run on to a different engine or no engine for each of the integration instance settings.</p> <ul style="list-style-type: none"><li>• Create load-balancing groups</li></ul> <p>When selecting Load-Balancing Group → Add to new group, you can create multiple load-balancing groups and decide which engines are part of each group.</p> <p>Users can move an engine from one group to another. A group will be deleted when the last engine is removed from it.</p> <p>Each engine can only belong to one group.</p>
Upgrade Engine	Relevant for Shell installation only. If you didn't install an engine using the Shell installation, you will need to remove the engine and do a fresh installation. For more information, see Upgrade an engine.
Get Logs	Logs are located in <code>/var/log/demisto</code> . For multiple engines, logs are located in <code>/var/log/demisto/&lt;name of the engine&gt;</code> . For example, <code>var/log/demisto.d1_e1</code> .
Edit Configuration	Relevant for Shell installation only. Enables you to edit the <code>d1.conf</code> file without having to access the file on your remote machine. For more information, see Configure engines.
Download Configuration	Download the <code>d1.conf</code> file to view the attribute values.
Delete Engine	Deletes an engine from Cortex AgentiX. To remove the engine from your remote machine, see Remove an engine.

### 3.6.5 | Upgrade an engine

#### Abstract

Upgrade an engine on Cortex AgentiX or directly on the remote machine.



Whenever there is a Cortex AgentiX major version change or a change in tenant-engine protocol version, your engines require an upgrade. On the Engines page, the Status column shows those engines that require upgrades. You can upgrade an engine by doing the following:

- If you installed the engine using the Shell installer, you can upgrade the engine on the Engines page.
- If you didn't install the engine using the Shell installer, you need to remove the engine and do a fresh install.

#### Upgrade an engine (shell installations)

You can upgrade the engine on the Engines page if you have installed the engine using the shell installer. The engine must be connected during the upgrade.

##### Customize upgrade variables

Before upgrading, we recommend you review the upgrade variables and verify if any need to be set in the `/usr/local/demisto/upgrade.conf` file on the engine. For environments with multiple engines, the file is located at `/usr/local/demisto/<engine-name>/upgrade.conf`. In some cases, usually related to a web proxy server or a custom directory, if you do not configure the `upgrade.conf` file, the upgrade will fail.

The option to set custom upgrade variables is only available for shell installation.

Variable	Description	Default
https_proxy	The URL of a web proxy server to use when connecting with the server. The variable name is case sensitive. Other common proxy variables, such as <code>http_proxy</code> or <code>HTTPS_PROXY</code> are ignored. Use <code>https_proxy</code> even if your proxy address begins with <code>http://</code> .	Not set
SERVER_URLS	The URL to connect to for hash validation. Set this variable if your tenant's address has changed. Use your tenant's API address, with the <code>api-</code> prefix added, instead of the UI address. For example: <code>SERVER_URLS="api-example.us.paloaltonetworks.com"</code> . Include only the IP/hostname and, optionally, a port. Do not include <code>https://</code> or any path at the end.	Public tenant URL
TRUST_ANY_CERTIFICATE	Determines whether the connection's SSL certificate must be trusted. This variable must be empty "" to require certificate trust. When set to <code>-k</code> , trusts any certificate. We recommend enabling this setting. Verify first that the engine host has the required CA root certificate, especially if using a proxy.	<code>-k</code>
XSOAR_ENGINE_AUTO_UPGRADE_TMP_DIR	Specifies a directory to use for extracting upgrade files and executing the upgrade. For example, <code>XSOAR_ENGINE_AUTO_UPGRADE_TMP_DIR="/root/tmp/engine1"</code> For environments with multiple engines, each engine must use a different temporary directory. This variable must be set if you used the <code>--target</code> option in the shell installer.	By default, a random directory under the <code>/tmp</code> folder is used.

##### Test upgrade connectivity

1. Test the upgrade connectivity by creating a mock `d1_upgrade.sh` file :

```
cd /usr/local/demisto  
echo test > d1_upgrade.sh
```

After you create the file, the upgrade cron job removes the file within one minute.



- Check the upgrade log file `/var/log/demisto/demisto_install.log` for connection-related errors. For hosts with multiple engines, the log file can be found at `/tmp/<engine name>/demisto_install.log`.
- If the test is successful, the following message appears at the end of the log file, with a recent timestamp: `Validation HTTPS request returned: false.`
- If you find errors in the log, you may need to change the variables in the `upgrade.conf` file or to change your network configuration.

#### How to upgrade

- On the Engines page, select the checkbox for the engine that requires an upgrade.
- Click Upgrade Engine.

When the upgrade finishes, the version appears in the Cortex AgentiX Version column. The upgrade procedure can take several minutes.

#### Upgrade an engine (non-shell installations)

If you didn't use the Shell installer, you need to remove the engine and do a fresh install.

- On the Engines page, locate the engine that requires an update.
- In the Download link, click the relevant Download files.
- On the remote machine, do the following:
  - Remove the existing engine. For more information, see Remove an engine.
  - Install the engine you downloaded in step 2. For more information, see Install an engine.

When the upgrade finishes, the version appears in the Cortex AgentiX Version column. The upgrade procedure can take several minutes.

#### Related information

Troubleshoot engines.

### 3.6.6 | Remove an engine

#### Abstract

Remove an engine by running the relevant command, depending on your operating system.

You can remove an engine when it is no longer needed.

- Run one of the following commands according to your operating system:

Installation	Command
RPM	Get the full package: <code>rpm -qa   grep -i ^d1_*</code> Remove the package: <code>rpm -evv d1_ &lt;package name&gt;</code>
DEB	Get the full package: <code>dpkg-query -W -f='\${Package}' d1_*</code> Remove the package: <code>dpkg --purge &lt;package name&gt;</code>
SH	Remove an Engine: <code>sudo &lt;engine-file-path&gt; -- --purge</code>

### 3.6.7 | Configure engines

#### Abstract

Configure Cortex AgentiX engines by editing the `d1.conf` file or modifying the configuration in the UI (for shell installations).

When installing an engine, a `d1.conf` file is installed on your machine. Some configurations can only be done by editing the `d1.conf` file. If you install via Shell, you can edit the configuration in the UI as well as edit the file directly.

A use case for modifying the engine configuration is if you want to generate engine logs for a specific log level.



#### Edit the d1.conf file

1. On the machine on which you installed the engine, navigate to the **d1.conf** file:

Installation Type	Location
RPM, DEB, Shell	/usr/local/demisto  If using multiple engines, the location is /usr/local/demisto/ <i>name of the engine</i> . For example, /usr/local/demisto/d1_e1
ZIP	Same folder as the binary.

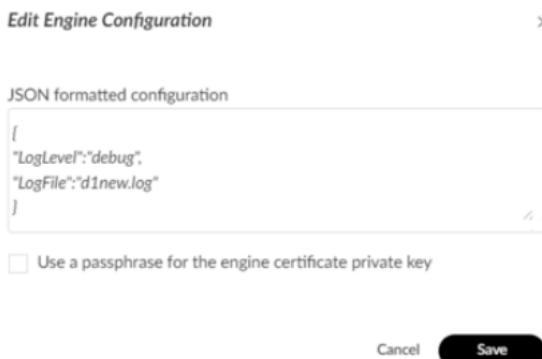
2. Modify the file as required. See Common properties when editing an engine configuration

You can also Configure the engine to use a web proxy.

#### Modify the configuration in Cortex AgentIX (Shell installations only)

Ensure that the data is in JSON format. The properties that you specify override the values defined in the **d1.conf** file.

1. From the engines table, select the engine for which you want to modify the configuration.
2. Click Edit Configuration.
3. In the JSON formatted configuration dialog box, modify the properties as required. For more information, see Common properties when editing an engine configuration.



#### Common properties when editing an engine configuration

The following table describes the common properties when editing an engine configuration using the **d1.conf** file (located by default at `/usr/local/demisto/`) or in the JSON formatted configuration dialog box in Cortex AgentIX.

Property	Type	Values	Edit
<code>http_proxy</code>	String	The IP address of the HTTP proxy through which the engine communicates.  For example, see Configure the engine to use a web proxy.	The engine <b>d1.conf</b> file.
<code>https_proxy</code>	String	The IP address of the HTTP/s proxy through which the engine communicates.  For example, see Configure the engine to use a web proxy.	The engine <b>d1.conf</b> file.



Property	Type	Values	Edit
LogLevel	String	<ul style="list-style-type: none"> <li>• debug</li> <li>• info</li> <li>• warning</li> </ul>	The engine <code>d1.conf</code> file or in the JSON formatted configuration dialog box.
<code>log.rolling.maxfilesize</code>	String	The maximum size in MB to retain log files. Default is 20 MB.	The engine <code>d1.conf</code> file.
<code>log.rolling.backups</code>	String	The maximum number of log files to retain. Default is 3.	The engine <code>d1.conf</code> file.
<code>log.rolling.maxage</code>	String	<p>The maximum number of days to retain old log files based on the time stamp encoded in the file name. Default is 0 (not to retain old log files based on age).</p> <p><b>NOTE:</b> A day is defined as 24 hours and may not exactly correspond to calendar days due to daylight savings, leap seconds, etc.</p>	The engine <code>d1.conf</code> file.
BindAddress	String	The port on which the engine listens for agent connection requests and communication task responses.	The engine <code>d1.conf</code> file.
EngineURLs	String array	An array of tenant addresses to which the engine tries to connect. If you change the tenant URL, you need to update this parameter.	<p>The engine <code>d1.conf</code> file.</p> <p><b>NOTE:</b> In addition, to support engine upgrades from the UI, edit the <code>/usr/local/demisto/upgrade.conf</code> file on the engine to include the <code>SERVER_URLS</code> setting with the new tenant's address. Include only the host, without <code>https://</code> or any additional path at the end of the host name. For example: <code>SERVER_URLS="api-example.us.paloaltonetworks.com"</code></p>
LogFile	String	Path to the <code>d1.log</code> file. If you change the name or location of the <code>d1.log</code> file, you need to update this parameter.	The engine <code>d1.conf</code> file.
<code>engine.handshake.max.retries.slow</code>	String	The maximum time in minutes the engine will try to reconnect after losing communication. Default is 600 (10 hours).	The engine <code>d1.conf</code> file.
		<p><b>NOTE:</b> If the engine loses communication for longer than this time, it will disconnect and you need to restart the service.</p>	



### 3.6.7.1 | Configure the engine to use a web proxy

#### Abstract

Configure a Cortex AgentX engine to use a web proxy by editing the `d1.conf` file.

Proxy settings can be configured in an engine by adding them as an engine configuration.

#### NOTE:

You need to configure Docker to use a proxy. When using a BlueCoat proxy, ensure you encode the values correctly.

1. On the machine on which you installed the engine, navigate to the `d1.conf` file and add the following keys.

Key	Value	Description
<code>http_proxy</code>	<code>http://&lt;user:password@proxy-server:port#&gt;</code> For example <code>http://user:password@proxy-server:3128</code>	Environment uses HTTP proxy. Special characters must be escaped.
<code>https_proxy</code>	<code>https://&lt;user:password@proxy-server:port#&gt;</code> For example, <code>https://user:password@proxy-server:3128</code>	Environment uses HTTPS proxy. Special characters must be escaped.
<code>no_proxy</code>	<code>http://&lt;user:password@proxy-server:port#&gt;</code> For example <code>http://user:password@proxy-server:3128</code>	For specific addresses, a proxy bypass will be applied. Special characters must be escaped.

2. If the environment variables are not set, or you wish to use different settings than those specified in the environment variables, set the configuration with your specific proxy details in the `d1.conf` file. For example:

```
{"http_proxy": "http://proxy.host.local:8080",
"https_proxy": "https://proxy.host.local:8443"
"no_proxy": "https://proxy.host.local:8020"}
```

3. Save the file.

4. On the machine where you installed the engine, navigate to the `upgrade.conf` file and edit the file to set `https_proxy` to your proxy address. For example, `https_proxy="https://proxy.host.local:8443"`.

#### NOTE:

In an environment with a single engine, go to `/usr/local/demisto/upgrade.conf`. In an environment with multiple engines, go to `/usr/local/demisto/<engine-name>/upgrade.conf`, replacing `<engine-name>` with the name of the engine.

Note that the key is in the `upgrade.conf` file and must be `https_proxy`, even if your proxy address starts with `http://`.

5. Save the file.

### 3.6.7.2 | Configure the engine to call the server without using a proxy

#### Abstract

Configure an engine to call the server without using a proxy.

In some cases, due to specific environment architecture, you may need to configure the engine to use a proxy when working with integrations, but not use a proxy when calling the Cortex AgentX tenant.

1. On the computer where you have installed the engine, go to the directory for `d1.conf` file.

For RPM, DEB, Shell go to `/usr/local/demisto`.

2. Add the following configuration:



Key	Value
<code>engine.to.server.proxy</code>	<code>false</code> (default is <code>true</code> )

### 3.6.7.2.1 | Use NGINX as a reverse proxy

#### Abstract

Use NGINX as a reverse proxy to the Cortex AgentiX engines.

NGINX can act as a reverse proxy that sits between internal applications and external clients, forwarding client requests to the appropriate application. Using NGINX as a reverse proxy in front of the engine enables you to provide network segmentation where the proxy can be put on a public subnet (DMZ) while the engine can be on a private subnet, only accepting traffic from the proxy. Additionally, NGINX provides a number of advanced load balancing and acceleration features that you can utilize.

If you want to use an engine (d1) through the reverse proxy, you need to modify `EngineURLs` in the `d1.conf` file to point to the host and port the NGINX server is listening on. In addition to supporting engine upgrades from the UI, edit the `/usr/local/demisto/upgrade.conf` file to add the `SERVER_URLS` setting. `SERVER_URLS` should be set to the proxy's network address (host and port). For example: `SERVER_URLS="10.0.0.30:1234"`. For `SERVER_URLS`, include only the IP/hostname and, optionally, a port. Do not include `https://` or any path at the end.

#### Install NGINX

You can install NGINX on the Red Hat/Amazon (yum) and Ubuntu Linux distributions. For full instructions and available distributions, see NGINX documentation.

1. On the engine, run one of the following commands according to your Linux system:

- RedHat/Amazon: `sudo yum install nginx`
- Ubuntu: `sudo apt-get install nginx`

2. (Optional) Verify the NGINX installation by running the following command:

```
sudo nginx -v
```

#### Generate a certificate for NGINX

You should not use self-signed certificates for production systems. It is recommended to use a properly signed certificate for production systems. These instructions are intended only for non-production setups.

1. To use OpenSSL to generate a self-signed certificate, on the engine machine, run the following command:

```
sudo openssl req -x509 -nodes -days 3650 -newkey rsa:2048 -keyout /etc/nginx/cert.key -out /etc/nginx/cert.crt
```

2. When prompted, complete the on-screen instructions to complete the required fields.

#### Configure NGINX

1. Open the following NGINX configuration file with your preferred editor:

```
/etc/nginx/conf.d/demisto.conf
```

2. Use the following configuration template:

Replace `DEMISTO_ENGINE` with the appropriate hostname.

```
# Replace DEMISTO_ENGINE with the appropriate hostname. If needed, change port 443 to the port on which the engine is listening.

upstream demisto {
    server DEMISTO_ENGINE:443;
}

# Uncomment to redirect http to https (optional)
# server {
#     listen 80;
#     return 301 https://$host$request_uri;
# }

server {
    # Change the port if you want NGINX to listen on a different port
    listen 443;

    ssl_certificate      /etc/nginx/cert.crt;
    ssl_certificate_key  /etc/nginx/cert.key;

    ssl on;
    ssl_session_cache builtin:1000 shared:SSL:10m;
    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
```



```

ssl_ciphers HIGH:!aNULL:!eNULL:!EXPORT:!CAMELLIA:!DES:!MD5:!PSK:!RC4;
ssl_prefer_server_ciphers on;

access_log /var/log/nginx/demisto.access.log;

location / {
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;

    proxy_pass https://demisto;
    proxy_read_timeout 90;
}

location ~ ^/(websocket|d1ws|d2ws) {
    proxy_pass https://demisto;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_set_header Host $host;
    proxy_set_header Origin "";
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
}
}

```

#### **NOTE:**

For multi-tenant deployments, replace `location ~ ^/(websocket|d1ws|d2ws) {` with `location ~ ^/(acc_\$+/)?(websocket|d1ws|d2ws)`

3. Restart the NGINX server by typing the following command:

```
sudo service nginx restart
```

4. Verify you can access the engine by browsing to the NGINX server host.

### 3.6.7.3 | Configure an engine to use custom certificates

#### Abstract

Replace the self-signed certificate for an engine with a valid CA certificate for communication tasks.

You can replace the default self-signed certificate for the engine with your own certificate.

1. Find the two files created by the engine. The default location is `/usr/local/demisto`.

```
d1.key.pem
```

```
d1.cert.pem
```

2. Replace the contents of these files with your own certificates.

3. Change file owner to demisto:

```
chown -R demisto:demisto d1.key.pem
```

```
chown -R demisto:demisto d1.cert.pem
```

4. Set the file permissions:

```
chmod 600 d1.key.pem
```

```
chmod 644 d1.cert.pem
```

### 3.6.8 | Use an engine in an integration

#### Abstract

Use an engine or a load-balancing group of engines to fetch issues and run commands for an integration.

When you create an integration instance, you can select whether to fetch issues and run commands executed for the integration using the engine or a load-balancing group of engines. After you add the engine or load-balancing group to an integration instance, you can run commands using the engine or load-balancing group by specifying the `using` argument in the Issues War Room.

Before configuring an integration to run using multiple engines in a load-balancing group, we recommend that you test the integration using a single engine in the load-balancing group.



#### **NOTE:**

Long-running integrations should not run on load-balancing groups.

#### Command Example

```
!url url="www.cnn.com" using=urlscan.io_instance_1
```

### 3.6.9 | Run a script using an engine

#### Abstract

Run a script on an engine or load-balancing group to distribute the workload and improve performance.

You can run a script on an engine or load-balancing group to distribute the workload and improve performance.

1. Go to Investigation & Response → Automation → Scripts.
2. Select the script and click Settings.
3. From the BASIC section, in the Run on field, select either a single engine or a load-balancing group.  
The option to select an engine or load-balancing group only appears if at least one engine or load-balancing group is connected.
4. From the list, select the name of the engine or load-balancing group.
5. Click Save.

### 3.6.10 | Troubleshoot engines

#### Abstract

Troubleshoot engines by accessing logs and viewing errors.

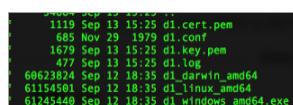
When troubleshooting engines, access the logs from Settings → Configurations → Engines and select the engine from which you want to download the logs.

#### **NOTE:**

Ensure that pop-ups are not blocked by your browser.

#### Debug engines

The d1.log field appears whenever an engine is running. The d1.log field contains information necessary for your customer success team to debug any engine-related issue. The field displays any error, as well as noting whether the engine is connected.



```
1119 Sep 13 15:25 dl.cert.pem
685 Nov 29 1979 dl.conf
1679 Sep 13 15:25 dl.pem
477 Sep 13 15:25 dl.log
60623824 Sep 12 18:35 dl_darwin_amd64
61154581 Sep 12 18:35 dl_linux_amd64
61245448 Sep 12 18:35 dl_windows_amd64.exe
```

#### Troubleshoot engine installation

#### **NOTE:**

If the installer fails to start due to a permissions issue, even if running as root, add one of the following two arguments when running the installer:

- --target <path> - Extracts the installer files into the specified custom path.
- --keep - Extracts the installer files into the current working directory (without cleaning at the end).

If using installer options such as -- -tools=false, the option should come after the --target or --keep arguments. For example:

```
sudo ./d1-installer.sh --target /some/temp/dir -- -tools=false
```

If you set a custom path when you run the installer, you must also set a custom path for upgrading your engine or the upgrade will fail. For more information, see [Upgrade an engine](#).

After installing the engine, check that the engine is connected to the Cortex AgentIX tenant and that it is running.

1. Go to Settings → Configurations → Engines and verify that the engine is connected.
2. If the engine is not connected, run the following command on the engine server to check if the engine service is running.

```
sudo systemctl status d1
```

#### **NOTE:**

If the Allow running multiple engines on the same machine option is selected, run the command:



```
sudo systemctl status d1_<Engine _name>
```

3. Access the d1 log on the engine server.

```
sudo tail -f /var/log/demisto/d1.log
```

- If the engine service is not running, and there's nothing relevant in the log, run `journalctl` on the engine server to understand why the installation failed.
- If the engine service is running, review the errors to see if the engine is failing to connect or if there are other issues (ignore all errors related to `\d2ws`, because this is not the same as `d1ws`.) Most often, the server address is incorrect and you will see an error like this:

```
error Cannot connect to [wss://<mainServerIP/HostName>/d1ws]: wss://<mainServerIP/HostName>/d1ws: dial tcp:  
lookup localhost: no such host. . Waiting 3 seconds. Will try untilâ€|
```

In this case, navigate to `/usr/local/demisto/d1.conf` and change the `EngineURLs` parameter to an address the engine can reach. Check the addresses at the beginning of the `upgrade_engine.sh` file. If the addresses are not correct, set the correct addresses in the `/usr/local/demisto/upgrade.conf` file, as a comma-separated list.

The configurations that might affect the `upgrade_engine.sh` script are the following variables are located at the beginning of the script:

- `SERVER_URLS`
- `TRUST_ANY_CERT`

If you make a change to the `baseURLs` configuration, you must apply the change in `/usr/local/demisto/d1.conf` AND in `/usr/local/demisto/upgrade.conf` under the `SERVER_URLS` var. For `SERVER_URLS`, specify only the IP/hostname and, optionally, a port. Do not include `https://` or any path at the end.

If you make a change in the `engine.connection.trust_any_certificate` configuration, you must apply the change in `/usr/local/demisto/upgrade.conf` as follows:

- If the `engine.connection.trust_any_certificate` configuration was set to true (trust any certificate), set the `TRUST_ANY_CERT` variable to `-k`.
- If the `engine.connection.trust_any_certificate` configuration was set to false, the `TRUST_ANY_CERT` variable should be blank (â€“).

#### **NOTE:**

Any changes made to variables in the `upgrade_engine.sh` file are reset after each upgrade. We recommend instead using the `upgrade.conf` file to set variables.

#### **NOTE:**

You can ignore the following error: `Cannot create folder '/var/lib/demisto'`

4. To check the connectivity from the engine to the Cortex AgentIX tenant, see *Troubleshoot engine connectivity* below.

5. If the installation issue remains, open a support case with logs from the engine.

- a. On the engine server, in `/usr/local/demisto/d1.conf`, set "LogLevel": "debug".
- b. Restart the d1 service and let it run for a few minutes.

```
sudo systemctl restart d1
```

#### **NOTE:**

If the Allow running multiple engines on the same machine option is selected, run the command:

```
sudo systemctl status d1_<Engine _name>
```

- c. Capture a `journalctl`:

```
journalctl --since "1 day ago" > engineTroubleshootingJournalctl.log
```

- d. On the engine server, tar up the log, conf, `journalctl`, and install log on the engine.

```
tar -cvzf engineLogs.tar.gz /var/log/demisto /usr/local/demisto/d1.conf /tmp/demisto_install.log  
engineTroubleshootingJournalctl.log
```

Troubleshoot engine upgrades

During an upgrade, the upgrade file is sent to the engine server. A cron job running on the engine server checks if that file exists. The most common upgrade error is that the job is not running, so the new installer does not run.

#### **NOTE:**

If the installer fails to start due to a permissions issue, even if running as root, add one of the following two arguments when running the installer:



- **--target <path>** - Extracts the installer files into the specified custom path.
- **--keep** - Extracts the installer files into the current working directory (without cleaning at the end).

If using installer options such as **--tools=false**, the option should come after the **--target** or **--keep** arguments. For example:

```
sudo ./d1-installer.sh --target /some/temp/dir -- tools=false
```

1. SSH to the machine.

2. Check the d1 service status on the engine server. It is possible that it stopped or doesn't exist.

```
sudo systemctl status d1
```

**NOTE:**

If the Allow running multiple engines on the same machine option is selected, run the command:

```
sudo systemctl status d1_<Engine _name>
```

3. Access the installer log on the engine server and review the error.

```
sudo vi /tmp/demisto_install.log
```

4. Rerun the installer on the engine using one of the following options. You can open a second window and run **watch df -h**. If the problem seems to be disk space, you should resolve the disk space issue and then rerun the installer.

5. Do one of the following:

- Download the installer from the user interface and copy it to the engine.

Add the following commands:

```
sudo chmod +x installer.sh
sudo ./installer.sh -- -y
```

- Verify that **/usr/local/demisto/d1\_upgrade.sh** exists.

1. Run the following commands:

```
sudo chmod +x /usr/local/demisto/d1_upgrade.sh
sudo /usr/local/demisto/d1_upgrade.sh
```

2. If **d1\_upgrade.sh** doesn't exist, check if **/usr/local/demisto/archived\_d1\_upgrade.sh** exists and that it was created at the time of the attempted upgrade.

3. If the file exists and was created at the time of the attempted upgrade, run the following commands on the engine server:

```
sudo chmod +x /usr/local/demisto/archived_d1_upgrade.sh
sudo /usr/local/demisto/archived_d1_upgrade.sh
```

## Troubleshoot engine connectivity

The following provides instructions for troubleshooting connectivity issues from the engine to the endpoint.

1. Follow the instructions in network troubleshooting.
2. Ensure that the engine can reach the endpoint by running the following command on the server engine.

```
sudo curl -kvv <endpointURL>
```

3. If the engine could not reach the endpoint, try the IP with curl instruction adding the http(s)://, or try using ping.

If this works, add the IP to the /etc/hosts file with the hostname and try to reach the endpoint again by running the following command on the engine server

```
sudo curl -kvv <endpointURL>
```

If this still fails, then this is an issue of connectivity between the engine and endpoint and you need to resolve this with your networking team.

4. After connectivity has been confirmed via curl:



- Try connecting within Docker without passing host networking.

```
docker run -it --rm demisto/netutils:1.0.0.6138 curl -kvv <endpointURL>
```

If this succeeds but the integration still fails, it could be an integration credentials issue. In that case, open a support case.

- If, without passing the host networking fails, run the following:

```
docker run -it --rm --network=host demisto/netutils:1.0.0.6138 curl -kvv <endpointURL>
```

If this succeeds, add "python.pass.extra.keys": "--network=host" to /usr/local/demisto/d1.conf and retest the integration.

If you see a Docker or SELinux issue, see Troubleshoot Docker networking issues .

5. If the installation issue remains, open a support case with logs from the engine.

- a. On the engine server, in /usr/local/demisto/d1.conf, set "LogLevel": "debug" .

- b. Restart the d1 service and let it run for a few minutes.

```
sudo systemctl restart d1
```

**NOTE:**

If the Allow running multiple engines on the same machine option is selected, run the command:

```
sudo systemctl status d1_<Engine _name>
```

- c. Capture a journalctl:

```
journalctl --since "1 day ago" > engineTroubleshootingJournalctl.log
```

- d. On the engine server, tar up the logs, conf, journalctl, and install log on the engine.

```
tar -cvzf engineLogs.tar.gz /var/log/demisto /usr/local/demisto/d1.conf /tmp/demisto_install.log  
engineTroubleshootingJournalctl.log
```

#### Engine 443 error

This error might occur when a connection is established between an engine and the Cortex AgentX tenant, because, by default, Linux does not allow processes to listen on low-level ports.

#### Error Message

```
listen tcp :443: bind: permission denied
```

#### Solution

- In the d1.conf file, change the port number to a higher one, for example, 8443.
- Run this command: sudo setcap CAP\_NET\_BIND\_SERVICE=+eip /path/to/binary. After running this command the server should be able to bind to low-numbered ports.

#### Bad handshake error

This error can occur in the engine logs relating to a bad handshake on the engine trying to connect to a Cortex AgentX tenant.

#### Error Message

```
Cannot connect to [wss:/xxx]: [wss://xxx|wss://xxx/]: websocket: bad handshake
```

#### Solution

Verify that time is synchronized on the engine to a reliable NTP source. When timing is off on the engine, this can cause a failure during the SSL/TLS handshake process. When time is resynced, connectivity from the engine to the parent server should be restored.

### 3.6.11 | Troubleshoot integrations running on engines

The following are common errors that occur when integrations are running on an engine.

#### Troubleshoot engine import error or invalid syntax error

When running an integration on an engine, the most common errors are:



- Broken Pipe
- "ImportError: No module named..."
- Invalid syntax
- Script failed to run: exec: "python": executable file not found in \$PATH (2603)

These errors could indicate that the engine is not using Docker.

1. Use SSH to access the engine server.

2. Make sure Docker is healthy.

- a. Ensure that Docker is installed and is running.

```
sudo systemctl status docker
```

If the Docker status is not good, restart your Docker.

```
sudo systemctl restart docker
```

- b. Ensure Docker can run a container.

```
sudo docker run hello-world
```

If this fails, reinstall your Docker.

3. Access the d1.conf file on the engine server.

```
sudo vi /usr/local/demisto/d1.conf
```

4. Add the "python.engine.docker": true configuration to the d1.conf file and remove any other configurations related to python and Docker, such as "python.executable.no.docker".

5. Restart the system on the engine server.

```
sudo systemctl restart d1
```

#### **NOTE:**

If the Allow running multiple engines on the same machine option is selected, run the command:

```
sudo systemctl restart d1_<Engine _name>
```

6. Retest the integration from the user interface. This may take a few minutes because it may need to pull the relevant Docker image.

Troubleshoot permission denied

A common error message you may see when running integrations on engines is something like: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get http://%2Fvar%2Frun%2Fdocker.sock/v1.35/images/json?t.

1. Determine if you are using a Docker group or Dockerroot group by running one of the following on the server engine:

- `ls -la /var/run/docker.sock`

The output from this command will show what user/group is running docker.sock. For example:

```
srw-rw----. 1 root docker 0 Apr 12 20:32 /var/run/docker.sock
```

shows that it's a Docker group and not Dockerroot.

- `cat /etc/group | grep docker`

This command shows if you are running Docker or Dockerroot.

#### **NOTE:**

Docker CE installations typically run Docker, while Docker EE installations typically run dockerroot.

2. To fix a Docker user, run the following commands on the server engine:

- a. `sudo groupadd docker`

- b. `sudo usermod -aG docker demisto`

- c. `sudo systemctl restart docker`

- d. `sudo systemctl restart d1`

#### **NOTE:**

If the Allow running multiple engines on the same machine option is selected, run the command:



```
sudo systemctl restart d1_<Engine _name>
```

3. To fix a dockerroot user, run the following commands on the server engine:

- a. `sudo groupadd dockerroot`
- b. Set the dockerroot group in `/etc/docker/daemon.json`. For example: `{ "group": "dockerroot" }`.
- c. `sudo usermod -aG dockerroot demisto`
- d. `sudo chcon -Rt svirt_sandbox_file_t /var/lib/demisto/temp`
- e. `sudo systemctl restart docker`
- f. `sudo systemctl restart d1`

**NOTE:**

If the Allow running multiple engines on the same machine option is selected, run the command:

```
sudo systemctl restart d1_<Engine _name>
```

## 3.7 | Cases and issues configuration

Learn how to configure cases and issues in Cortex AgentiX.

### Abstract

Configure cases and issues in Cortex AgentiX including domains, fields, and layouts. Set up correlation rules and configure an external dynamic list.

#### 3.7.1 | Case and issue lifecycle

### Abstract

Cases and issues go through various processes in Cortex AgentiX including ingestion, case/issue creation, planning, investigation, and response.

Cortex AgentiX uses the following for structured containers for all actionable security issues:

- Issues

Issues identify the problems that you need to solve in your environment. Cortex AgentiX creates issues when problems occur in your environment that cross defined thresholds or surpass your organization's accepted level of risk and threat tolerance. Issues are the primary focus of autonomous actions.

There are several issue triggers, including:

- SIEM issues
- Mail issues
- Security issues

These issues are generated from third-party services, such as SIEMs, mailboxes, and data.

- Cases

Cases are a workbench for resolving security problems in your environment. Each case groups related issues, highlights the impacted assets, and provides essential data in one place. Cases help you stay focused on threats and risks that have the most impact on the organization's security, help you reduce noise in your environment, and guide you to resolution using automation actions that reduce time and effort.

Cases serve as a broader, overarching container for complex or prolonged security investigations, capable of encompassing multiple related issues over time, providing a holistic view and serving as a central point for intricate or multi-stage security incidents.

For more information about cases and issues in Cortex AgentiX, see Investigation and Response.

Cortex AgentiX includes several out-of-the-box cases and issue types, fields, and layouts, some of which can be customized to suit your use case. You can also create custom fields and layouts as necessary. Cases can be created manually, from an API, or from an integration feed.

You can define integrations with your third-party security and incident management vendors. You can trigger events from these integrations that become cases/issues in Cortex AgentiX. You can run playbooks on these cases/issues to enrich them with information from other products in your system, which helps you complete the picture.

### Planning

Before you begin configuring integrations and ingesting information from third parties, consider the following:



Phase	Description
Data ingestion	<p>Data ingestion is the entry point of all security data into Cortex AgentiX, which relies on integrations to pull in alerts, events, and data from third-party sources. Configure integrations with third-party products to start fetching events, such as potential phishing emails, authentication attempts, and SIEM events. For more information, see Content configuration.</p>
Create case/issue domains	<p>Consider whether you want to create case domains for your use case. Every organization has unique incident response plans, escalation matrices, and remediation steps. Customizing cases allows you to align the platform's workflow precisely with your established internal processes. For example, a cyber incident response domain or a compliance and risk management domain. SOC analysts focus on active threats (cyber incident response), and GRC teams handle long-term risk (compliance and risk management).</p> <p>For more information, see Case and issue domains.</p>
Fields and layouts	<p>Customize Cortex AgentiX security investigations by tailoring precisely to an organization's specific operational needs, workflows, and reporting requirements. This ensures the platform truly reflects the unique context of each SOC.</p> <p>SOC engineers can create and manage unique data fields essential for capturing specific information (for example, business impact rating, affected Department, and design the visual presentation of an issue or case).</p> <p>Cortex AgentiX relies on these definitions to understand what information to collect, where to store it, and how to present its findings, while human analysts interact with these structured views for efficient collaboration and oversight.</p> <p>For more information, see Customize your cases.</p>
Playbooks	<p>Configure playbooks for automated handling, enrichment, and disposition of issues once they are created. Set up triggers to run a playbook on an issue. For more information, see Automations.</p>

### 3.7.2 | Customize your cases

#### 3.7.2.1 | External integrations

##### Abstract

Gain additional verification on key artifacts by integrating Cortex AgentiX with other Palo Alto Networks and third-party security products.

You can integrate external threat intelligence services with Cortex AgentiX that provide additional verification sources for each key artifact in a case. Cortex AgentiX supports the following integrations:

##### Threat intelligence

Integration	Description
WildFire	<p>Cortex AgentiX automatically includes WildFire threat intelligence in the case and issue investigation.</p> <p>WildFire detects known and unknown threats, such as malware. The WildFire verdict contains detailed insights into the behavior of identified threats. The WildFire verdict is displayed next to relevant Key Artifacts in the Cases page. See Review WildFire analysis details for more information.</p>



Integration	Description
VirusTotal	<p>VirusTotal provides aggregated results from over 70 antivirus scanners, domain services included in the block list, and user contributions. The VirusTotal score is represented as a fraction. For example, a score of 34/52 means out of 52 queried services, 34 services determined the artifact to be malicious.</p> <p>To view VirusTotal threat intelligence in cases, you must obtain the license key for the service and add it to the Cortex AgentiX Configuration. When you add the service, the relevant VirusTotal (VT) score is displayed in the Cases page under Key Artifacts.</p>

### 3.7.2.2 | Create a case domain

#### Abstract

You can create custom case domains to help you to differentiate between your work efforts, and effectively manage and prioritize your workload.

#### WARNING:

Before you add a custom domain, please review the built-in options. For more information, see [Case and issue domains](#).

We recommend using the built-in domains where possible. Custom domains might not be supported by all content. In addition, custom domains affect Cortex AgentiX's ability to learn, correctly identify, and score future cases.

Smart grouping and SmartScore are not supported for custom domains.

Custom domains help you to differentiate between your work efforts. You can create tailored workflows for each domain, so that you can effectively manage and prioritize your workload.

#### Manage your domains

You can see all domains under Configurations → Object Setup → Cases → Domains. From this tab, you can edit the properties of the built-in domains and create your own custom domains.

Consider the following information:

- You can't merge cases with different domains.
- SmartScore and smart grouping are not supported for custom domains.
- For SBAC, use the Cases and Issues scoping area to define case and issue domains that enable you to control access to your domains. For more information, see [Manage user scope](#).
- Domains might affect custom content that is connected to cases and issues. Review your custom content to ensure it is associated with the intended domains. This includes:
  - Automation Rules
  - Starring Rules
  - Notifications
  - Issue Exclusions
  - Scoring Rules
  - XQL that accesses the cases or issues datasets in Scheduled Queries and Widgets

#### How to create a case domain

#### NOTE:

- Adding custom domains requires a View/Edit RBAC permission for Case Properties (under Object Setup).
- Once created, a custom case domain cannot be deleted or renamed.

1. Go to Settings → Configurations → Object Setup → Cases → Domains .

The existing domains are listed.

2. Click on + New Domain.

3. Assign a name and color to the domain, and an optional description.



4. In the Status field, select one or more statuses that are relevant to the domain. These statuses will be available for selection in the cases and issues associated with this domain.
5. In the Resolution Type field, select one or more resolution reasons that are relevant to the domain. These reasons will be available for selection in the cases and issues associated with this domain.
6. Click Save.
7. (Optional) Update SBAC scoping to enable access to the domain.
  - a. You can perform the following:
    - To enable access to the domain for a User Group, go to Settings → Configurations → Access Management → User Groups.
    - To enable access to the domain for a User, go to Settings → Configurations → Access Management → Users.
    - To enable access to the domain for an API key, go to Settings → Configurations → Integrations → API Keys.
  - b. When editing an existing User Group, User, or API key, in the Scope tab you can update the granular scoping for the new Endpoints domain.
  - c. Click Save.

### 3.7.2.3 | Create a starring configuration

You can proactively star issues and the cases to which they are linked by creating a starring configuration:

1. Select Cases & Issues → Case Configuration → Starred Issues.
2. Select Add Starring Configuration.
3. Under Configuration Name, enter a name to identify your starring configuration.
4. (Optional) Under Comment, enter a descriptive comment.
5. In the issue table, use the filters to define the issue attributes you want to include in the match criteria. For example, you can select issues with High severity, issues by category, or issues associated with certain assets or asset providers.

**TIP:**

Right-click an issue field to add it as match criteria.

6. Click Create.

#### Scope-Based Access Control considerations

Case starring supports Scope-Based Access Control (SBAC). The following parameters are considered when editing a starring configuration:

- If Scope-Based Access Control (SBAC) is enabled and the Endpoint Scoping Mode is set to restrictive mode, you can edit a configuration if you are scoped to all tags in the configuration.
- If Scope-Based Access Control (SBAC) is enabled and the Endpoint Scoping Mode is set to permissive mode, you can edit a configuration if you are scoped to at least one tag listed in the configuration.
- If a policy was added when set to restrictive mode, and then changed to permissive (or vice versa), you will only have view permissions.

### 3.7.2.4 | Create custom case statuses and resolution reasons

#### Abstract

You can create custom case status and resolutions that are tailored to your workflow.

**NOTE:**

Before you create a custom status, please review the built-in options. For more information, see [Resolution reasons for cases and issues](#).

We recommend using the built-in statuses and resolution reasons where possible. Custom statuses and resolution reasons might not be supported by all content, and status syncing can take time.

In addition, custom statuses affect Cortex Agent's ability to learn, correctly identify, and score future cases.

You can create custom cases statuses and custom resolution reasons that are tailored to your workflow. Custom case statuses and resolution reasons apply to case and issue statuses, and can also be used in playbooks.

Adding custom case statuses and resolution reasons requires a View/Edit RBAC permission for Case Properties (under Configurations → Object Setup).

**NOTE:**



**After creation, custom statuses and resolution reasons cannot be deleted or modified.**

How to create custom case statuses

1. Go to Configurations â Object Setup â Cases â Properties.

The existing statuses and resolution types are listed.

2. In the Add another status field, type a new status and click Save.

3. Click Edit to rearrange the order of the statuses. This order is presented when you set a status or select a resolution type.

### 3.7.3 | Customize issue fields and layouts

Abstract

You can create custom issue fields and custom issue layouts for out-of-the-box and custom issue fields.

You can create custom issue fields and custom issue layouts. Custom issue layouts can include both out-of-the-box and custom issue fields. Issue layouts are applied to issues according to layout rules.

#### 3.7.3.1 | Issue fields

Abstract

Add issue fields for mapping, correlation rules, issue custom layouts and for display in the issues table.

Cortex AgentIX includes out-of-the-box issue fields, issue fields from installed content packs, and user defined custom issue fields. You can use issue fields for mapping, correlation rules, and custom issue layouts.

All system and custom issue fields are available in the Issues table. New custom fields are hidden by default. To show custom issue fields in the Issues table, click the three dot vertical ellipses and select the column(s) from the list.

For Grid fields, HTML fields, and Markdown fields, if the field contains data the Issues table shows Data Available instead of the values. To view the data, open the issue and click Investigate to see the full issue layout. For multi-select fields, the first value is shown in the Issues table and the number of additional values is stated, but the additional values are not shown. For example, if a multi-select field holds the values x, y, and z, the Issues table shows x + 2 More.

Cortex AgentIX stores both the original value of the field and the current value of the field, if different. Any changes made between the original value and the current value are not stored. For example, if the original value of the field was x, the value was then changed to m, and then changed to y, only the x and y values are stored. To view the original value and the current value of changed fields, hover over the updated issue fields icon  on the right side of the row in the Issues table. To revert all of the fields in an issue to their original values, click Restore all fields to their original values in the updated issue fields box. Restoring all fields to their original values also restores the original values in the issue context data. Once you restore fields to their original values, this action can not be undone.

Custom issue fields can be exported and imported. To export a single custom issue field, right-click on the field in the fields table, and select Export. To export all custom issue fields in a single JSON file, click the Export All button above the fields table. System issue fields cannot be exported or imported.

After a custom issue field is created, it can be edited, deleted, or exported by right-clicking on the row. The field name and field type cannot be changed after the field is created. System fields cannot be edited, deleted, or exported.

#### **WARNING:**

Deleting an issue field or uninstalling a content pack containing an issue field may affect detection and other capabilities based on the deleted field. For example, correlation, layouts, case scoring, starring rules, and playbook triggers.

#### 3.7.3.1.1 | Issue field types

Abstract

When creating issue fields, you can add field types, such as Boolean, date picker, and grid (table).

You can create the following types of issue fields.

Read more...



Field	Description
Boolean	<p>True or False</p> <ul style="list-style-type: none"> <li>• Incoming values <code>0</code>, <code>false</code>, and <code>False</code> are treated as False.</li> <li>• Incoming values <code>true</code>, <code>True</code>, or any number besides 0 are treated as True.</li> <li>• Other values are treated as null.</li> </ul>
Date picker	<p>Adds the date to the field.</p> <p>Supported time formats for validation are ISO 8601 and Epoch. Other values are treated as null.</p> <p><b>NOTE:</b></p> <p>You cannot set filters, starring rules, automation rules, layout rules, or issue exclusions based on the values in custom timestamp fields.</p>
Grid (table)	<p>Include an interactive, editable grid as a field. For details, see Create a grid field for an issue.</p> <p><b>NOTE:</b></p> <p>When grid field is shown in the Issues table, if there are values in the field, they do not display in the table. Instead, the column shows Data Available.</p>
HTML	<p>Create and view HTML content.</p> <p><b>NOTE:</b></p> <p>When an HTML field is shown in the Issues table, if there is a value in the field, it does not display in the Issues table. Instead, the column shows Data Available.</p> <p>The following HTML tags are not permitted: <code>blockquote</code>, <code>del</code>, <code>dd</code>, <code>div</code>, <code>dl</code>, <code>dt</code>, <code>fieldset</code>, <code>form</code>, <code>h1</code>, <code>h2</code>, <code>h3</code>, <code>h4</code>, <code>h5</code>, <code>h6</code>, <code>hr</code>, <code>iframe</code>, <code>ins</code>, <code>li</code>, <code>math</code>, <code>noscript</code>, <code>ol</code>, <code>pre</code>, <code>p</code>, <code>script</code>, <code>style</code>, <code>table</code>, <code>ul</code>, <code>address</code>, <code>article</code>, <code>aside</code>, <code>canvas</code>, <code>details</code>, <code>dialog</code>, <code>figcaption</code>, <code>figure</code>, <code>footer</code>, <code>header</code>, <code>hgroup</code>, <code>main</code>, <code>nav</code>, <code>output</code>, <code>progress</code>, <code>section</code>, <code>video</code>.</p> <p>The following CSS tags are not permitted: <code>background-color</code>, <code>text-align</code>, <code>font-size</code>, <code>font-family</code>, <code>font-weight</code>, <code>color</code>, <code>line-height</code>, <code>border-style</code>, <code>border</code>, <code>page-break-inside</code>, <code>tablelayout</code>, <code>padding</code>, <code>background-size</code>, <code>display</code>, <code>padding-top</code>, <code>padding-right</code>, <code>padding-bottom</code>, <code>padding-left</code>, <code>text-size-adjust</code>, <code>break-inside</code>, <code>word-break</code>, <code>width</code>, <code>height</code>, <code>-ms-text-size-adjust</code>, <code>-webkit-text-size-adjust</code>.</p>
Long text	<ul style="list-style-type: none"> <li>• Long text is analyzed and tokenized, and entries are indexed as individual words, enabling you to perform advanced searches and use wildcards.</li> <li>• Long text fields cannot be sorted and cannot be used in graphical dashboard widgets.</li> <li>• While editing a long text field, pressing enter will create a new line. Case is insensitive.</li> </ul>
Markdown	<p>Add markdown-formatted text as a Template which is displayed to users in the field. Markdown lets you add basic formatting to text to provide a better end-user experience.</p> <p><b>NOTE:</b></p> <p>When a Markdown field is shown in the Issues table, if there is a value in the field, it does not display in the table. Instead, the column shows Data Available.</p>
Multi select / Array	<p>Includes two options:</p> <ul style="list-style-type: none"> <li>• Multi select from a pre-filled list.</li> <li>• An empty array field for the user to add one or more values as a comma-separated list.</li> </ul> <p>In the Basic Settings section, enter a comma-separated list of values.</p>
Number	<p>Can contain any number. Default is 0.</p>



Field	Description
Short Text	<ul style="list-style-type: none"> <li>Short text is treated as a single unit of text, and is not indexed by word. Advanced search, including wildcards, is not supported.</li> <li>Short text fields are case insensitive.</li> <li>While editing a short text field, pressing enter will save and close.</li> <li>Recommended use is one word entries. Examples: username, email address, etc.</li> </ul>
Single select	Select one from a list of options. Add a list of comma-separated values. By default, the first value is used, unless the checkbox for Use first as default is cleared.
Timer	Timer fields enable you to view how much time has passed since the timer was started and how much time remains until the timer times out. You can also configure a script to run when a timer times out.
URL	Contains a URL.

#### 3.7.3.1.2 | Create custom issue fields

##### Abstract

Create issue fields so you can map from incoming issues, map the output of queries from correlation rules, and add them to custom issue layouts.

You can create custom issue fields to:

- Map raw JSON fields from incoming issues.
- Display custom fields data in the Issues table.
- Create correlation rules that generate issues from XQL queries and map the output of the queries to custom issue fields.
- Design custom issue layouts that include custom issue fields.

How to create a custom issue field:

1. Select Settings → Configurations → Object Setup → Issues → Fields → New Field.

2. Choose a field type and enter a field name. For a description of available field types, see Issue field types. You can add an optional tooltip to provide users with information about the field.

If adding a grid, see Create a grid field for an issue.

3. Click Save.

Custom issue fields can be exported and imported. To export a single custom issue field, right-click on the field in the fields table, and select Export. To export all custom issue fields in a single JSON file, click the Export All button above the fields table.

After a custom issue field is created, it can be edited, deleted, or exported by right-clicking on the row. The field name and field type cannot be changed after the field is created.

You can also update the custom field values by running the Set command in the CLI, a script, or a playbook. For more information, see Update issue fields.

#### 3.7.3.1.2.1 | Create a grid field for an issue

##### Abstract

You can add a grid field to a custom issue layout.

The grid field enables you to view and edit a table. You can add a grid field to a custom issue layout.

1. Select Settings → Configurations → Object Setup → Issues → Fields → New Field.

2. In the New Issue Field window under Field Type, select Grid (table).

3. Complete the following parameters:



Parameter	Description
Field Name	Name for the grid field.
Tooltip	(Optional) A brief descriptive message that explains what the field is and how to use it.
User can add rows	(Optional) Enables users to add/remove rows in the grid.

4. In the Grid tab, add or remove rows and columns.

How you design the grid determines how it appears to users. If you select user can add rows, the user can add rows but not columns.

5. Configure each column by selecting the required field types, such as short text, Boolean, URL, etc. You can move the columns, rename, and add values.

If you select the Lock check box, the value for that field is static (not editable). If you do not select the Lock check box (default), users can perform inline editing.

6. Click Save.

#### 3.7.3.1.2.2 | Issue field triggered scripts

##### Abstract

Associate Cortex AgentiX issue fields with scripts that are triggered when the field changes.

Issue fields can be assigned scripts that run when the field changes. This enables you to automate workflows during an issue lifecycle. These scripts can perform any action, such as dynamically changing the field value or notifying the responder when an issue severity has been changed. Field-triggered scripts can include conditions that must be met for the script to run, such as the field having a certain value.

Scripts can be created in Python, PowerShell, or JavaScript on the Scripts page. To use a script with a field trigger, you need to add the field-change-triggered tag to the script. You can then add the script in the Attributes tab when you edit or create an issue field. If you did not add the tag when creating the script, it cannot be selected until you add the tag.

When a script is associated with an issue field, changes to that field are saved only after the triggered script finishes running. This allows you, for example, to perform verifications such as checking that a specific field has been filled out before allowing a user to resolve an issue.

If you perform a bulk update and change the same field across multiple issues at the same time, and that field has a field-triggered script assigned, the script runs in each issue.

An issue field-triggered script can modify multiple issue fields. Note that if field A changes and a script is triggered and changes field B, and field B is also assigned a field-triggered script, the script for field B is not triggered.

Cortex AgentiX comes out-of-the-box with the emailFieldTriggered script, which sends an email to the issue owner when the selected field is triggered. You can also create your own custom scripts.

##### CAUTION:

This feature assumes fair and intended usage of field-triggered scripts. In cases of excessive or abusive usage, execution may be restricted or disabled. If script execution is restricted or disabled, fields are still updated, but without the results of the assigned script.

#### Issue field triggered script arguments

Issue field-triggered scripts have the following triggered field information available as arguments (args):

Argument	Description
<code>associatedToAll</code>	Whether the field is associated with all or some issues. Value: <code>true</code> or <code>false</code> .
<code>associatedTypes</code>	An array of the issue types with which the field is associated.



Argument	Description
<code>cliName</code>	The name of the field when called from the command line.
<code>description</code>	The description of the field.
<code>isReadOnly</code>	Specifies whether the field is non-editable. Value: <code>true</code> or <code>false</code> .
<code>name</code>	The name of the field.
<code>new</code>	The new value of the field.
<code>old</code>	The old value of the field.
<code>ownerOnly</code>	Specifies that only the creator of the field can edit. Value: <code>true</code> or <code>false</code> .
<code>placeholder</code>	The placeholder text.
<code>required</code>	Specifies whether this is a mandatory field. Value: <code>true</code> or <code>false</code> .
<code>selectValues</code>	If this is a multi-select type field, these are the values the field can take.
<code>system</code>	Whether it is a Cortex AgentiX defined field.
<code>type</code>	The field type.
<code>unmapped</code>	Whether it is not mapped to any issue.
<code>useAsKpi</code>	Whether it is being used for tracking KPI on an issue page.
<code>validationRegex</code>	Whether there is a regex associated validation for the values the field can hold.

**NOTE:**

Fields that can hold a list, such as multi-select custom fields, return the delta in an array as a new argument. For example, if a multi-select field value has changed from `["a"]` to `["a", "b"]`, the new argument of the script gets a value of `["b"]`.

Add an issue field triggered script to an issue field

After creating an issue field-triggered script in the Scripts page in Python, PowerShell, or JavaScript, you can then associate it with an issue field.

1. Go to Settings â Configuration â Object Setup â Issues â Fields.

2. Right-click the issue field and select Edit.

3. In the Attributes tab, under Script to run when field changes, select the desired issue field-triggered script.

**NOTE:**



Issue field-triggered scripts must have the `field-change-triggered` tag to appear in the list.

Issue field trigger scripts are not supported for all system fields. The following fields are not supported for issue field trigger scripts and may result in failing to populate the issue layout:

- **Cases:** Case ID, Cases IDs
- **Asset Fields:** Asset IDs, Asset Names, Asset Classes, Asset Categories, Asset Groups, Asset Regions, Asset Providers, Asset Accounts, Asset Types
- **Other:** Business Application Names, Findings

Field-change-triggered with single select or multi select types

1. Create and save a single select or multi-select script in the Scripts page.

**NOTE:**

When creating the script, add the `field-change-triggered` tag in the script settings.

Example 10.

This is an example of a single select script.

```
# Mapping of user selection to email addresses
owner_mapping = {
    'option1': 'alice@example.com',
    'option2': 'eled@example.com',
    'option3': 'carol@example.com',
    'option4': 'dave@example.com',
    'option5': 'eve@example.com',
}

# The value selected by the user when the script is triggered
val = demisto.args().get('new')

# Get the mapped email address
owner_email = owner_mapping.get(val, val)

# Set the owner of the incident
demisto.executeCommand('setIssue', {
    'owner': owner_email
})
```

2. Go to Settings → Configurations → Object Setup → Issues → Fields.

3. Click New Field and create a new issue field of one of the following types:

- Single select
- Multi-select

4. Click Basic Settings and in the Values section set the values you want to see in the issue layout dropdown list for this field.

For example, `option1,option2,option3,option4,option5`.

5. Click Attributes and in Script to run when field changes, select the script you created in Step 1.

6. Go to Settings → Configurations → Object Setup → Issues → Layouts and add the new issue field to an existing layout or create a new layout.

7. In the issue layout edit page, click Fields and Buttons and drag the new issue field you created to the layout.

8. Save the version.

9. Select one of the values. The layout will update with the mapped value as set on the script related to the issue field.

Use scripts with a grid field

You can use scripts to manipulate and populate data in a grid field. In this example, analysts add comments to issues they work on during their shifts. The script automatically populates a column of the grid, logging the timestamp of each comment.

1. Create a script called `ShiftSummariesChange`. The script operates in the following phases:



- The script gets all new rows and sets the Date Logged field to now (current day).
- For each existing row, if the name matches, and the findings column is not updated, the Date Logged column is also updated.
- After creating a grid field, it is saved with the new values using the `setIssue` command.

```

var newField = args.new ? JSON.parse(args.new) : [];
//if line(s) added, set "datelogged" to now.
if (oldField.length < newField.length) {
    // for each new line change date.
    for(var i=oldField.length; i < newField.length; i++) {
        newField[i].datelogged = new Date().toISOString();
    }
}
var columnName = "findings";
// for each old line if the "columnName" has changed, change date to now.
for(var i=0; i < oldField.length; i++) {
    if (newField[i] && oldField[i].fullname === newField[i].fullname &&
        oldField[i][columnName] !== newField[i][columnName]) {
        newField[i].datelogged = new Date().toISOString();
    }
}
var newVal = {};
newVal[args.cliName] = newField;
executeCommand("setIssue", newVal);

```

2. Add the `field-change-triggered` tag and save the script.

3. Create a `Shift Summaries` grid field with the following columns:

- Full name
- Findings
- Status
- Date Logged

Select Date picker with the Lock checkbox, so the script can populate the values for that column. If a column is unlocked (default), the column values can be entered manually (by users), or by a script.

#### **NOTE:**

Verify that User can add rows is selected.

Add a row to a grid

During playbook execution, if a malicious finding is discovered, you can add that finding to a grid, using a script in a playbook task.

This Python script requires two arguments:

- `fieldCliName`: The machine name for the field for which you want to add a new row.
- `Row`: The new row to add to the grid. This is a JSON object in lowercase characters, with no white space.

```

fieldCliName = demisto.args().get('field')
currentValue = demisto.incidents()[0]["CustomFields"][fieldCliName];

if currentValue is None:
    currentValue = [json.loads(demisto.args().get('row'))]
else:
    currentValue.append(json.loads(demisto.args().get('row')))

val = json.dumps({ fieldCliName: currentValue })
demisto.results(demisto.executeCommand("setIssue", { 'customFields': val }))

```

3.7.3.1.2.3 | Issue timer fields

Abstract

Issue timer fields count up from when a specific event begins and can also count down to a deadline. You can trigger actions in the event the issue timer field is breached.

By default, timer fields are disabled in Cortex AgentIX. To enable timer fields, go to Settings → Configurations → General → Server Settings → Issues and Enable Timer Field.

Timer issue fields provide you with the ability to track reaction time and help you measure issue-level metrics. Timers can measure multiple aspects of an issue. You can, for example, have a timer track how long since the first playbook ran, and have another timer track how long you've been waiting for a user's response. Timers display in the issues table and in issue layouts.

Timer fields can be started, stopped, or paused in a playbook, script, or manually in the CLI.



Timer fields count up from when a specific action or task began and also (optionally) count down from a target. The Risk Threshold tells you when a timer is considered at risk and you can customize the time period for the Risk Threshold.

Timer fields always show the total duration while they are still running. If they are at risk, they show the at risk status. After a timer field has timed out (passed the target), the timer shows both the total duration and how long past the target.

Timer fields do not automatically trigger actions when timers time out. You can configure a script to run when a timer times out.

#### Scripts

You can run scripts to act on timeouts, such as sending an email when a timeout occurs. You can also make specific changes to an issue field or a parent case issue, such as changing the case owner. Cortex AgentX includes out-of-the-box scripts or you can create your own scripts. Scripts must have the **SLA** tag to be used for timer fields. For more information, see Automate changes to issue fields using timer scripts.

#### Using the CLI

If you want to set or change timers for an issue you can use the **setIssue** command in the CLI. You can also use commands such as **startTimer**, **stopTimer**, and **pauseTimer**. For more information, see Use issue timer field commands manually in the CLI.

#### Configure issue timer fields

##### Abstract

Create a timer fields and optionally add scripts to trigger when timers have been breached.

You can create timer fields that display in the issues table and issue layouts. When you create an issue timer field, you have the option of providing a target for completion and also the option of triggering a script when the timer field has timed out (the target has passed).

If you set a target for a timer field, the Risk Threshold is automatically activated and displays when the timer is considered at risk. You can customize the timeframe for the Risk Threshold. If you do not provide a target, the timer only counts up from when it was triggered.

You can start, stop, or pause a timer from the CLI, from scripts, and from playbooks.

##### How to create a timer issue field

1. Go to Settings â—» Configurations â—» Object Setup â—» Issues â—» Fields â—» New Field.
2. Select Timer as the Field Type.
3. Type a field name.
4. (Optional) Under Basic Settings, Timer you have the option of setting a target for the timer field. By default, the timer field shows hours and minutes. You can change this to days and hours, by clicking Hours. If you do not enter the number of hours and minutes, the timer only counts up from when it is triggered.

If you set a target in the timer field, by default the Risk Threshold field is activated. You can edit the Risk Threshold value.

5. (Optional) Under Run script on timeout, select the script to run when the target has timed out. For example, you could write a script that sends an email when the target has timed out. For more information, see Automate changes to issue fields using timer scripts.

##### NOTE:

Only scripts to which you have added the **SLA** tag appear in the list of scripts you can select. To add a tag to a script, create a new script or edit an existing script and enter the tag name in the script settings.

When you hover over the machine name (below the Field Name) note the name which is used in the command line or script.

6. Save the field.

7. (Optional) Add the field to one or more issue layouts. By default, the timer field is available to view in the issues table.

#### Configure a playbook to run timers

##### Abstract

Add or configure a playbook to run timers.

Within a playbook, you can set a timer to start, pause, or stop at a specific section header or task. For example, you can create a timer called Pending user response and have it start in a playbook when an email is sent to a user. If the user does not respond within the target timeframe, then you can automatically send an additional reminder to the user or run a different task.

To select a timer in a task or section header, in the Timers tab select the action that you want the timer to perform for the task. You can add multiple timers to a task or section header, so in the same task you can stop one timer and start another.

##### NOTE:

When a task or section has a timer configured, it displays the hourglass icon.



The following table describes the timer options:

Option	Description
<code>Timer.start</code>	<p>Starts the timer.</p> <p><b>NOTE:</b></p> <p>Timers are not started automatically when a case is created.</p>
<code>Timer.pause</code>	Pauses the timer. A paused timer can be started again without being reset.
<code>Timer.stop</code>	<p>Stops the timer. Information about the timer is still displayed in the issue layout and/or issues table, but the status displays as Ended.</p> <p><b>NOTE:</b></p> <p>If you stop a timer before the issue is closed, you must reset the timer using the <code>resetTimer</code> command before you can start the timer again. When you reset the timer, all fields are cleared.</p>

Some playbooks, such as Phishing - Generic v3, come out-of-the-box with timer tasks included. If you need the same timers across use cases, create a sub-playbook based on your use case or conditions such as issue severity.

If you want to stop or pause a timer in a playbook, you can use an existing task or create a new section header/task. When you select Timer.stop, the run is considered finished and cannot be restarted without setting it to zero. If you plan to restart the timer, select Timer.pause so you do not lose the accumulated time. By default, all timers stop when the case closes.

Automate changes to issue fields using timer scripts

Abstract

Create scripts to perform specific actions in Cortex AgentiX when a timer field times out

Scripts in Cortex AgentiX enable you to automate processes. You can create scripts that perform specific actions when a timer field times out. Scripts used with timers must have the SLA tag.

You can use an out-of-the-box or custom script and attach it to an timer issue field.

A common use of scripts for timer fields is to send an email when a timer is breached. You can create a custom script that sends an email to specific users when the script is triggered. You can add this to any timer issue field as needed.

Use issue timer field commands manually in the CLI

Abstract

Set and change timers for specific issues, such as decreasing the required response time for a high-priority issue.

You can manage the timers for a specific issue by running commands manually in the CLI. By running CLI command you can to manage timers on a more granular level within specific issues when the need arises. For example, for a high severity issue you might need to decrease the response time.

Set timer fields

Use the `setIssue` command to set a specific issue due date, or to set a specific timer field in an issue. If you add the `sla` parameter to the command, it sets the time for the issue's due date. If you also add the `slaField` you set the timer for the issue field.

Example 11.

To change the Time to Assignment field target to 30 minutes in the current issue, run the following command:

```
!setIssue sla=30 slaField=timetoassignment
```

To change the timer to February 1, 2024, at 11.12 am, run the following command:

```
!setIssue sla=2024-02-01T11:12
```

**NOTE:**

When defining the values for the `slaField` use the machine name for the field, which is lowercase and without spaces. You can check the machine name by editing the issue field.



Start or stop timer fields

Run the following commands in the CLI:

Command	Description
<code>startTimer</code>	<p>Starts the timer.</p> <p>This command can also be used to restart a paused timer.</p> <p>Example 12.</p> <pre>!startTimer timerField=timetoassignment</pre> <p><b>NOTE:</b></p> <p>Timer fields are not started automatically when an issue is created unless run in a playbook.</p>
<code>pauseTimer</code>	<p>Pauses the timer.</p> <p>Use this command when a timer field has already started.</p> <p>Example 13.</p> <pre>!pauseTimer timerField=timetoassignment</pre>
<code>stopTimer</code>	<p>Stops the timer.</p> <p>Example 14.</p> <pre>!stopTimer timerField=timetoassignment</pre> <p><b>NOTE:</b></p> <p>After a timer field is stopped, before you can start the timer again you must reset the timer using the <code>resetTimer</code> command.</p> <p>Timers are automatically stopped when an issue is closed.</p>
<code>resetTimer</code>	<p>Clears all fields for the timer.</p> <p>This command must be used before restarting a timer that was stopped.</p> <p>Example 15.</p> <pre>!resetTimer timerField=timetoassignment</pre>

**NOTE:**

When running commands in the CLI, you can specify the `alertID` to change the timer for a different issue.

### 3.7.3.2 | Issue layouts

Abstract

View system layouts or create custom layouts for your issue type.

Cortex AgentIX includes default issue layouts. You can add additional issue layouts by installing content packs, duplicating system issue layouts, or creating new custom issue layouts. Issue layouts are applied to incoming issues based on issue layout rules.

Issue layouts control the information displayed in the Investigate panel. To see the issue layout that has been applied, in the Issue investigation panel click the Layout Info button  in the upper right corner. Empty layout fields are hidden by default, but are shown if you select Show empty fields.

The default issue layouts and any layouts that are added from content packs, are locked by default and cannot be deleted, edited, or exported. To view a system layout, right-click the layout row and select View. If you want to edit a system layout, you can detach or duplicate the layout by right-clicking the layout row in the issue layout table and selecting Detach or Duplicate. If you detach a layout, the layout does not receive content updates until it is reattached. To reattach a system layout, right-click the layout row and select Attach. If you detach a layout and make changes, those changes may be overwritten if you later



reattach the layout. If a layout is detached, you can edit or duplicate it, but you cannot delete or export it. If you instead duplicate the issue layout, the new duplicated layout can be edited, deleted, or exported, the same as a custom issue layout.

When viewing an issue, most issue fields can be edited inline, by users with editing permissions. After editing a field inline, click the check mark to save your change. Some system fields, such as Source Instance, cannot be edited.

To modify an existing custom layout, go to Settings â€“> Configurations â€“> Object Setup â€“> Issues â€“> Layouts, right-click the layout in the layouts table, and select Edit, Duplicate, Delete, or Export.

### 3.7.3.2.1 | Create custom issue layouts

#### Abstract

Create an issue layout to select the specific fields and buttons you require.

Custom issue layouts let you choose the specific fields and buttons that are displayed for different types of issues. You can create custom issue layouts that include both custom and out-of-the-box issue fields, and add buttons with tasks that can assist and guide analysts in their investigation.

You can import a custom issue layout by clicking Import and uploading the JSON file. You can also modify or task actions on an existing layout existing layout by right-clicking the layout in the layout table.

#### Create a custom issue layout

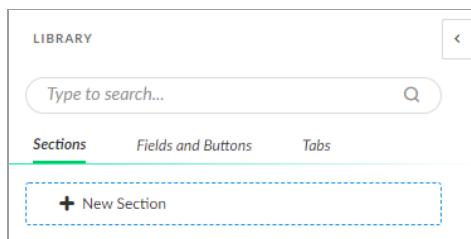
1. Go to Settings â€“> Configurations â€“> Object Setup â€“> Issues â€“> Layouts â€“> New Layout.
2. Enter a name for the layout.
3. (Optional) Add new tabs to the layout, and drag them to change the order in which they appear.

The Issue Info tab and any new tabs you create can be renamed, hidden, duplicated, or deleted. Hover over the tab name and click the settings button to see the available options. You cannot edit or delete the War Room and Work Plan tabs in the issue layout, but you can hide them by clicking the settings button, and selecting Hide tab.

By default, empty fields within the tab are hidden in the issue layout. To show empty fields, hover over the tab name, click the settings button, and select Show empty fields.

4. Add new sections to the Issue Info tab, or click +New tab.

To add a new section, from the Sections tab of the Library drag a New Section into a tab. You can also add the predefined sections, such as Malicious or Suspicious Indicators and War Room Entries.



5. Customize the section

Clicking the pencil icon for a section, to configure how a section appears. You can hide or show the section header, and configure the section fields to appear in rows or as cards.

Some sections have additional configuration options. If you add a Malicious or Suspicious Indicators section, you can configure an indicator search query. If you add a War Room Entries section, you can filter by type of entry, such as chats, notes, or files.

The General Purpose Dynamic Section enables you to configure a section that displays the results of a script. Only scripts to which you have added the **dynamic-section** tag appear in the dropdown list. You can use the General Purpose Dynamic Section to display widgets, text, markdown, or HTML. For an example of how to add a widget with this section, see [Add a custom widget to an issue layout](#).

6. Add custom or out-of-the-box issue fields to the layout.

From the Fields and Buttons tab of the Library, drag fields into sections of the layout.

#### TIP:

Limit the number of issue fields to 50 in each section. You can create additional sections as needed.

7. Add buttons to the layout.

Buttons allow you to add tasks to your layout, which can assist an analyst. For example, you can add a button to scan a host or kill a process.

1. From the Fields and Buttons tab of the Library, drag a buttons into a section of the layout.
2. Click to configure.



3. Enter a descriptive name for the button, select a color, and select the script that you want to run when the button is clicked.

For fields (script arguments) that are optional, you can define whether to show them to analysts when they click on buttons. To expose an optional field, select the Ask User checkbox next to the script argument(s) in the button settings page.

**NOTE:**

The script that runs when an action button is clicked accepts only mandatory arguments through the pop up window and does not provide an option for any non-mandatory arguments to be filled in when the button is clicked. We recommend using a wrapper script to collect and validate arguments in scenarios where there can be a combination of mandatory and non-mandatory arguments for a button.

8. Save the layout.

Export issue layouts

You can export custom issue layouts and duplicates of system issue layouts

To export a single issue layout, right-click on the layout in the layouts table, and select Export. To export all custom issue layouts and duplicates of system issue layouts in a single JSON file, click the Export All button above the layouts table.

3.7.3.2.2 | Add a custom widget to an issue layout

Abstract

Add any widget to a custom issue layout.

You can add a custom or system widget to a custom issue layout by uploading an auto script and using it in a General Purpose Dynamic Section in your layout.

Example 16.

The following example shows how to add an Indicator Widget Bar. This custom widget script shows the severity of indicators in an issue, as a bar chart.

1. Add the Indicator Widget Bar script to Cortex AgentIX.

a. Go to Investigation & Response → Automation → Scripts and upload the following script:

Expand script

```
commonfields:
  id: ee3b9604-324b-4ab5-8164-15ddf6e428ab
  version: 49
name: IndicatorWidgetBar
script: |-
  # Constants
  HIGH = 3
  SUSPICIOUS = 2
  LOW = 1
  NONE = 0

  indicators = []
  scores = {HIGH: 0, SUSPICIOUS: 0, LOW: 0, NONE: 0}
  incident_id = demisto.incidents()[0].get('id')

  foundIndicators = demisto.executeCommand("findIndicators", {"query": "investigationIDs:{}".format(incident_id), "size":999999})[0]['Contents']

  for indicator in foundIndicators:
    scores[indicator['score']] += 1

  data = {
    "Type": 17,
    "ContentsFormat": "bar",
    "Contents": [
      "stats": [
        {
          "data": [
            scores[HIGH]
          ],
          "groups": None,
          "name": "high",
          "label": "incident.severity.high",
          "color": "rgb(255, 23, 68)"
        },
        {
          "data": [
            scores[SUSPICIOUS]
          ],
          "groups": None,
          "name": "medium",
          "label": "incident.severity.medium",
          "color": "rgb(255, 144, 0)"
        },
        {
          "data": [
            scores[LOW]
          ],
          "groups": None,
          "name": "low"
        }
      ]
    ]
  }
```



```

        "name": "low",
        "label": "incident.severity.low",
        "color": "rgb(0, 205, 51)"
    },
    {
        "data": [
            scores[NONE]
        ],
        "groups": None,
        "name": "unknown",
        "label": "incident.severity.unknown",
        "color": "rgb(197, 197, 197)"
    }
],
"params": {
    "layout": "horizontal"
}
}
}

demisto.results(data)
type: python
tags:
- dynamic-section
enabled: true
scripttarget: 0
subtype: python3
runonce: false
dockerimage: demisto/python3:3.7.3.286
runas: DBotWeakRole

```

b. Click Save.

2. Add the widget to an issue layout.

a. Go to Settings → Configurations → Object Setup → Issues → Layouts.

b. Create a new custom issue layout or right-click to open an existing custom issue layout or a detached or duplicated system layout.

c. Drag and drop the General Purpose Dynamic Section into a layout tab.

d. Edit the General Purpose Dynamic Section by clicking the pencil icon.

e. Enter a name for the section and choose the automation script you uploaded in Step 1.

f. Click Ok.

### 3.7.3.2.3 | Create rules for issue layouts

#### Abstract

Add rules to assign a custom issue layout based on the issue source,

Issue layouts are applied to issues according to layout rules. Using a layout rule, you can assign a custom issue layout based on the issue source, such as a specific layout for issues generated from a correlation rule.

You can create multiple rules. If the first rule does not apply to the incoming issue, the next rule is checked, and so on. If a content pack is installed and it contains a layout rule, by default the layout rule is placed at the top of the rules list. You can change the order of the rules by dragging and dropping the rules in the list. You can filter the rule list by name, description, rule, layout, and source. If no layout rules apply to the issue, a default issue layout is used.

To edit or delete existing rules, right-click on the rule in the list and select Edit or Delete.

#### How to create layout rules

1. Go to Settings → Configurations → Object Setup → Issues → Layout Rules → New Rule.

2. Enter a rule name, select the layout to use if the rule is met, and provide a description.

3. Search for issues that match the criteria you want to use for the layout rule. For example, you can search for issues from a specific issue source.

4. Click Create.

5. Repeat as needed to create multiple rules.

6. Click Save.

#### SBAC considerations

Layout rules support SBAC (scoped based access control). The following parameters are considered for editing access.



- If Scope-Based Access Control (SBAC) is enabled and Endpoint Scoping Mode is set to restrictive mode, you can edit a rule if you are scoped to all tags in the rule.
- If Scope-Based Access Control (SBAC) is enabled and Endpoint Scoping Mode is set to permissive mode, you can edit a rule if you are scoped to at least one tag listed in the rule.
- As a scoped user who has editing permissions to a rule, you can change the order among other rules that are locked.
- If a rule was added when set to restrictive mode, and then changed to permissive (or vice versa), you will only have view permissions.

### 3.7.4 | Customize case fields and layouts

#### Abstract

You can create custom case fields and custom case layouts for out-of-the-box and custom case fields.

You can create custom case fields and custom case layouts. Custom case layouts can include both out-of-the-box and custom case fields. Case layouts are applied to cases according to layout rules.

#### 3.7.4.1 | Case fields

#### Abstract

Add case fields for custom case layouts and for display in the Cases table.

Cortex AgentIX includes out-of-the-box case fields, case fields from installed content packs, and user defined custom case fields. Case fields can be used for custom case layouts, and for display in the Cases table.

Custom case fields can be exported and imported. To export a single custom case field, right-click on the field in the fields table, and select Export. To export all custom case fields in a single JSON file, click the Export All button above the fields table. System case fields cannot be exported or imported.

After a custom case field is created, it can be edited, deleted, or exported by right-clicking on the row. The field name and field type cannot be changed after the field is created. System fields cannot be edited, deleted, or exported.

#### **WARNING:**

Deleting a case field or uninstalling a content pack containing a case field may affect capabilities based on the deleted field and layouts.

#### 3.7.4.1.1 | Case field types

#### Abstract

When creating case fields, you can select field types, such as Boolean, date picker, and grid (table).

You can create the following types of case fields:

Read more...

Field	Description
Boolean	<p>True or False</p> <ul style="list-style-type: none"> <li>• Incoming values <code>0</code>, <code>false</code>, and <code>False</code> are treated as False.</li> <li>• Incoming values <code>true</code>, <code>True</code>, or any number besides 0 are treated as True.</li> <li>• Other values are treated as null.</li> </ul>
Date picker	<p>Adds the date to the field.</p> <p>Supported time formats for validation are ISO 8601 and Epoch. Other values are treated as null.</p> <p><b>NOTE:</b></p> <p>You cannot set filters, starring rules, automation rules, layout rules, or issue exclusions based on the values in custom timestamp fields.</p>



Field	Description
Grid (table)	<p>Include an interactive, editable grid as a field. For details, see Create a grid field for a case.</p> <p><b>NOTE:</b></p> <p>When grid field is shown in the Cases table, if there are values in the field, they do not display in the Cases table. Instead, the column shows Data Available.</p>
HTML	<p>Create and view HTML content.</p> <p><b>NOTE:</b></p> <p>When an HTML field is shown in the Issues table, if there is a value in the field, it does not display in the Cases table. Instead, the column shows Data Available.</p>
Long text	<ul style="list-style-type: none"> <li>Long text is analyzed and tokenized, and entries are indexed as individual words, enabling you to perform advanced searches and use wildcards.</li> <li>Long text fields cannot be sorted and cannot be used in graphical dashboard widgets.</li> <li>While editing a long text field, pressing enter will create a new line. Case is insensitive.</li> </ul>
Markdown	<p>Add markdown-formatted text as a Template which is displayed to users in the field. Markdown lets you add basic formatting to text to provide a better end-user experience.</p> <p><b>NOTE:</b></p> <p>When a Markdown field is shown in the Cases table, if there is a value in the field, it does not display in the Cases table. Instead, the column shows Data Available.</p>
Multi select / Array	<p>Includes two options:</p> <ul style="list-style-type: none"> <li>Multi select from a pre-filled list.</li> <li>An empty array field for the user to add one or more values as a comma-separated list.</li> </ul> <p>In the Basic Settings section, enter a comma-separated list of values.</p>
Number	<p>Can contain any number. Default is 0.</p>
Short Text	<ul style="list-style-type: none"> <li>Short text is treated as a single unit of text, and is not indexed by word. Advanced search, including wildcards, is not supported.</li> <li>Short text fields are case insensitive.</li> <li>While editing a short text field, pressing enter will save and close.</li> <li>Recommended use is one word entries. Examples: username, email address, etc.</li> </ul>
Single select	<p>Select one from a list of options. Add a list of comma-separated values. By default, the first value is used, unless the checkbox for Use first as default is cleared.</p>
SLA	<p>SLA can be used to trigger a notification when the status affecting the SLA of a case changes. In this example, if the SLA is breached an email is sent to the owner's supervisor.</p> <p>For more information on SLAs, see Create case timers and SLAs.</p>
Timer	<p>Timer fields enable you to view how much time has passed since the timer was started and how much time remains until the timer times out. You can also configure a script to run when a timer times out.</p>
URL	<p>Contains a URL.</p>



**NOTE:**

If you make changes to case fields you can update the context data by running a playbook, script, or command. For more information, see [Update case fields](#).

To update dynamic custom case fields, such as SLA and Timer fields, see [Update case timer and SLA fields](#).

3.7.4.1.2 | [Create custom case fields](#)**Abstract**

Create case fields so you can add them to custom case layouts.

Create case fields so you add them to custom case layouts.

You can create custom case fields to:

- Map raw JSON fields from incoming issues.
- Display custom fields data in the Cases table.
- Create correlation rules that generate issues from XQL queries and map the output of the queries to custom case fields.
- Design custom case layouts that include custom case fields.

How to create a new custom case field:

1. Select Settings → Configurations → Object Setup → Cases → Fields → New Field.
2. Choose a field type and enter a field name. You can add an optional tooltip to provide users with information about the field.  
If adding a grid, see [Create a grid field for a case](#).
3. Save your changes.

Custom case fields can be exported and imported. To export a single custom case field, right-click on the field in the fields table, and select Export. To export all custom case fields in a single JSON file, click the Export All button above the fields table.

After a custom case field is created, it can be edited, deleted, or exported by right-clicking on the row. The field name and field type cannot be changed after the field is created.

3.7.4.1.2.1 | [Create a grid field for a case](#)**Abstract**

Create a grid field to add to a custom case layout.

Grid fields enable you to view and edit tables in a custom case layout.

1. Select Settings → Configurations → Object Setup → Cases → Fields → New Field.
2. In the New Case Field window Field Type field drop down list, select Grid (table).
3. Complete the following parameters:

Parameter	Description
Field Name	A meaningful name for the grid field.
Tooltip	(Optional) A brief descriptive message that explains what the field is and how to use it.
User can add rows	(Optional) Enables users to add/remove rows in the grid.

4. In the Grid tab, add or remove the required rows and columns.

How you design the grid determines how it appears to users. If the user can add rows field is selected, the user can add rows but not columns.

5. Configure each column by selecting the required field types, such as short text, Boolean, URL, etc. You can move the columns, rename, add values, etc.

If you select the Lock check box, the value for that field is static (not editable). If you do not select the Lock check box (default), users can perform inline editing.



6. Click Save.

#### 3.7.4.1.2.21 Update case fields

##### Abstract

Use a playbook, script, or command to update case fields.

Sometimes you need to update case fields based on a change in an issue. For example, after starting an investigation an analyst might want to change the name of a case, star a case, or change the status of a case.

You can update the following case fields through a playbook, script, or command:

- `manual_severity`
- `starred`
- `assigned_user_email`
- `status`
- `score`
- `incident_name`
- `description`

The following sections explain how to update case fields by running a command in the CLI, and running a script, and running a playbook.

##### Use the CLI

Run the `!setParentIncidentFields` command in the issue or case War Room.

When you start typing the CLI provides the available options. If you select an enum field the CLI provides the available values.

##### Examples

- To change the name of the case to `Malware`, run  
`!setParentIncidentFields incident_name=Malware`
- To change the name of the case to `Malware` and star the case, run  
`!setParentIncidentFields incident_name=Malware starred=true`

##### Use a script

When a script runs in an issue, the data from the script is added to the issue context data and the issue fields. If you want to update case fields, in a Json file, add the `setParentIncidentFields` to the `demisto.executeCommand` function.

##### Example

To update the case status to `resolved`, run

```
demisto.executeCommand("setParentIncidentFields", {"status": "resolved_other"})
```

##### NOTE:

Ensure that you have the required RBAC permission to write scripts.

##### Use a playbook

When running a playbook, by default the data is added to the issue context data and issue fields. You can additionally add this data to case context data and case fields by configuring tasks in a playbook.

The following example explains how to add tasks to a playbook that update the case fields to star a case, and add the key `starred: true` to the case context data.

1. Add the following tasks to a new or existing playbook.

- a. Create a Conditional task to check whether the parent incident fields are starred using the  `${parentIncidentFields.starred}`  key.



Standard  Conditional  Data Collection  Section Header

#4 Is not starred? (1)

Built-in  Manual  Ask  Choose script

Condition Details Timers Advanced On Error

Condition for: yes

[Remove condition](#)

Get  
\${parentIncidentFields.starre  
d}

Is empty  
(General)

+ And

IF ELSE IF

b. Create a standard task using the `setParentIncidentFields` script to update the starred field.

TASK DETAILS ✖

Standard  Conditional  Data Collection  Section Header

#5 set starred (1)

Script: `setParentIncidentFields (Builtin)` (1) ?

Inputs Outputs Mapping Advanced Details Timers On Error

incident\_name (1)

description (1)

manual\_severity (1)

Select from predefined values or add your own (1) ▾

starred (1)

true (1) ▾

assigned\_user\_mail (1)

status (1)

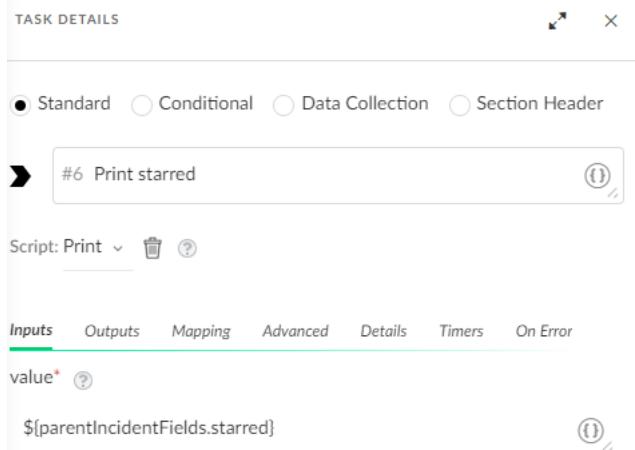
Select from predefined values or add your own (1) ▾

score (1)

Cancel OK

c. Create a standard task to print the value to the War Room.





## 2. Run the playbook.

In the case context data, you can see the key `starred: true`. If running in an issue or a case, after refreshing the case, the case is now starred.

### 3.7.4.1.3 | Create case timers and SLAs

#### Abstract

You can set up case timers and SLAs to track KPIs and ensure that operational performance is inline with your objectives. By adding timer and SLA fields to the Cases table, you can track the progress of your case SLAs.

To help you to monitor and assess your key performance indicators (KPIs), you can create SLAs at the case level. Case SLAs provide the ability to track KPIs, obtain real-time insights into operational performance, and ensure alignment with established objectives.

Case SLAs are based on case timer fields. When a case matches the defined criteria, the timer starts running. If the timer field is linked to an SLA, Cortex AgentiX tracks the progress of the case in relation to the SLA.

To track your SLAs on the Cases page, add the timer and SLA fields to the table layout, or create a custom layout with SLA fields. Note that the timer field counts forward, and the SLA field counts backwards.

#### PREREQUISITE:

Before you can create a case SLA, you must first create a timer field. A timer field can be associated with a single case SLA.

#### Create a case timer

Take the following steps to create a case timer field:

1. Go to Settings â Configuration â Object Setup â Case and open the Fields tab.
2. Click New Field.
3. Under Field Type, select Timer.
4. Type a field name.
5. Under Tooltip, enter a description to pop-up when you hover over the field.
6. Under Case Filter, click Set Filter and define the subset of cases for which the timer will be activated. For example, you can define timers for specific domains or case source types.

#### NOTE:

If you edit this filter after creation, the timer and associated SLA will be removed from any case that no longer qualifies, even if the timer is already running.

7. Under Conditions, add filters that define when the timer will start and end. To add a pause condition to the timer, click Pause and define the pause criteria.
8. Under When case is reopened, select the action that you want Cortex AgentiX to take.
9. Click Save.

#### Example 17.

The following timer measures the amount of time a security case is waiting in New status before an analyst starts investigating.



Field	Value
Field Type	Timer
Field Name	Security case response
Tooltip	Measure time from case opening to analyst response.
Cases Filter	Case Domain = Security
Start when	Status = New
End when	Status = Under Investigation
When case is reopened	Reset timer

#### Create a case SLA

Take the following steps to create a case SLA. You can set up multiple goals for an SLA.

1. Go to Settings â€“ Configurations â€“ Object Setup â€“ Cases and open the Fields tab.
2. Click New Field.
3. Under Field Type, select SLA.
4. Type a name to identify the SLA.
5. Under Tooltip, enter a description to pop-up when you hover over the field.
6. Under Timer, select the timer field with which to associate the SLA.
7. Under Goals, click Add SLA Goal.

The default goal applies to all cases that meet the filter criteria specified in the timer field. You can set up addition goals that apply to subsets of the defined cases.

8. In the SLA goal, type a goal name and set filter criteria.
9. In the Days, Hours, or Minutes fields, define the time conditions for to the SLA goal.
10. Arrange the SLA goals by dragging them in order of goal priority.
11. Click Save.

#### Example 18.

The following SLA field sets goals for analyst response times for security cases with Critical and High severity. This SLA is based on the timer field created in the previous example. Because the timer field is set up with the filter Case Domain = Security, this SLA will apply to security cases only.

The first SLA goal applies to security cases with a severity level of Critical. The SLA specifies that an analyst must respond to critical severity cases within one hour.

The second SLA goal applies to security cases with a severity level of High. The SLA specifies that an analyst must respond to high severity cases within two hours.

Field	Value
Field Type	SLA



Field	Value
Field Name	Security case response SLA
Tooltip	Measure time from case opening to analyst response.
Timer	Security case response
Goals	<ul style="list-style-type: none"> <li>Name: Critical severity cases</li> <li>Minutes: 60</li> <li>Filter: severity = Critical</li> </ul>
	<ul style="list-style-type: none"> <li>Name: High severity cases</li> <li>Minutes: 120</li> <li>Filter: severity = High</li> </ul>

Display timer and SLA fields in the Cases page

After creating new timer and SLA fields, you can add them to the Cases table layout and view them in the Cases detailed view:

- In the Cases table view, add the timer and SLA fields to the Layout tab in the Table Setting Menu.
- In the Cases detailed view, use the Sort By field to filter the cases list by the SLA field. Details of the SLA are shown in the list.

In addition, you can create a custom case layout with a new tab displaying SLA fields. For more information, see Case layouts.

Example of SLA and timer fields

#### Example 19.

This example is based on the fields created in the previous procedures:

- The Security case response timer field displays the number of minutes since case creation. When the case status moves from New to Under Investigation, the timer stops.
- The Security case response SLA field starts counting backwards to show the remaining time to meet the SLA. If the field is shown in red with a minus time, the SLA is breached.
  - For case 001, the critical severity case has been in New status for 5 minutes. An analyst must respond within the remaining 55 minutes.
  - For case 002, the high severity case has been in New status for 20 minutes. An analyst must respond within the remaining 1 hour and 40 minutes.
  - For case 003, an analyst did not respond within 60 minutes and therefore the SLA was breached. The Security case response SLA field displays a minus value and a red icon.

Case ID	Severity	Security Case Response	Security Case Response SLA
001	Critical	5m	55m 25s
002	High	20m	1h 40m 30s
003	Critical	65m	- 5m 23s



Consider the following information when working with timer and SLA fields:

- When a case is resolved, the timer calculation stops.
- Updating timer logic affects open and new cases. Therefore, the timer and associated SLA will be removed from any case that no longer qualifies, even if the timer is already running.
- If you delete a timer field, the SLA associated to the timer is also deleted.

#### 3.7.4.1.4 | Update case timer and SLA fields

Abstract

Update case timer and SLA fields by running the `RefreshIncidentDynamicCustomFields` command in the War room.

You can update case timer and SLA fields by running the following CLI command in the War Room:

```
!RefreshIncidentDynamicCustomFields
```

#### 3.7.4.2 | Case layouts

Abstract

View the different case layouts and create your own case layout.

Cortex AgentIX includes default case layouts. The default case layouts and any layouts that are added from content packs, are locked by default and cannot be deleted, edited, exported or duplicated.

You can add customized case layouts, of which you can Edit, Duplicate, Delete or Export. When creating or editing a case layout, you can only add/edit tabs that you created. You cannot edit the default tabs that exist in the layout.

#### 3.7.4.2.1 | Create custom case layouts

Abstract

Create a case layout to select the specific fields and buttons you require.

Custom case layouts let you choose the specific fields and buttons that are displayed for different types of cases. You can create custom case layouts that include both custom and out-of-the-box case fields.

Tabs that were created can be renamed, hid, duplicated, or deleted. To make these changes, hover over the tab name, click the settings button, and select the relevant option. You can drag and drop tabs to change the order they appear. By default, empty fields within the tab are hidden in the case layout. To show empty fields, hover over the tab name, click the settings button, and select Show empty fields.

You cannot edit the Overview, Key Assets & Artifacts, Issues & Insights, Timeline, War Room, and Executions tabs in the case layout. Select Hide Tab to hide the tab, rather than deleting the tab as you may want to use the tab again for future use.

Custom case layouts and duplicates of system case layouts, can be exported. To export a single case layout, right-click on the layout in the layouts table, and select Export. To export all custom case layouts and duplicates of system case layouts in a single JSON file, click the Export All button above the layouts table.

You can import a custom case layout by clicking Import and uploading the JSON file.

How to create a custom case layout

1. Select Settings → Configurations → Object Setup → Cases → Layouts → New Layout.
2. Enter a name for the layout.
3. To add a section, click on New or from the Library , under the Sections tab, drag and drop New Section into the new custom tab. You can also add a Notes section to the tab.

By clicking on the pencil icon for a section, you can configure how a section appears, by hiding or showing the section header, as well as configuring the section fields to appear in rows or as cards.

4. To add custom or out-of-the-box case fields to the layout, drag the fields from the Fields tab into existing sections or new sections that you added to the layout.

#### TIP:

Limit the number of case fields to 50 in each section. You can create additional sections as needed.

5. Add buttons to the layout.



Buttons allow you to add tasks to your layout, which can assist an analyst. For example, you can add a button to scan a host or kill a process.

1. From the Fields and Buttons tab of the Library, drag a buttons into a section of the layout.
2. Click to configure.
3. Enter a descriptive name for the button, select a color, and select the script that you want to run when the button is clicked.

For fields (script arguments) that are optional, you can define whether to show them to analysts when they click on buttons. To expose an optional field, select the Ask User checkbox next to the script argument(s) in the button settings page.

**NOTE:**

The script that runs when an action button is clicked accepts only mandatory arguments through the pop up window and does not provide an option for any non-mandatory arguments to be filled in when the button is clicked. We recommend using a wrapper script to collect and validate arguments in scenarios where there can be a combination of mandatory and non-mandatory arguments for a button.

For information on Filters and Transformers, refer to Filter and transform data.

6. Save the layout.
7. (Optional) To modify an existing layout, right-click the layout in the layout table and select Edit, Duplicate, Delete, or Export.

#### 3.7.4.2.1 Create rules for case layouts

##### Abstract

Add rules to assign a custom case layout based on the case source.

Case layouts are applied to case according to layout rules. For example, using a layout rule, you can assign a custom case layout based on the case, such as a specific layout for cases with a high severity.

You can create multiple rules. If the first rule does not apply to the incoming case, the next rule is checked, and so on. If a content pack is installed and it contains a layout rule, the layout rule is placed at the top of the rules list, by default. You can change the order of the rules by dragging and dropping the rules in the list. You can filter the rule list by name, description, rule, layout, and source. If no layout rules apply to the case, a default case layout is used.

To edit or delete existing rules, right-click on the rule in the list and select Edit or Delete.

**NOTE:**

Layout rules support SBAC (scoped based access control). The following parameters are considered for editing access.

- If Scope-Based Access Control (SBAC) is enabled and Endpoint Scoping Mode is set to restrictive mode, you can edit a rule if you are scoped to all tags in the rule.
- If Scope-Based Access Control (SBAC) is enabled and Endpoint Scoping Mode is set to permissive mode, you can edit a rule if you are scoped to at least one tag listed in the rule.
- As a scoped user who has editing permissions to a rule, you can change the order among other rules that are locked.
- If a rule was added when set to restrictive mode, and then changed to permissive (or vice versa), you will only have view permissions.

##### How to create rules for case layouts

1. Select Settings → Configurations → Object Setup → Cases → Layout Rules → New Rule.
2. Enter a Rule Name, select the custom or out-of-the-box Layout to Display if the rule is met, and provide a Description.
3. Search for cases that match the criteria you want to use for the layout rule. For example, you can search for cases from a specific case source.
4. Click Create.
5. Repeat as needed to create multiple rules.
6. Click Save.

#### 3.7.5 | Issue syncing

##### Abstract

Set up integrations that mirror Cortex issues with external applications, such as Jira or ServiceNow.

You can set up integrations in Cortex AgentX that mirror Cortex issues with external applications, such as Atlassian Jira or ServiceNow. When mirroring issues (also referred to as issue syncing), you can make changes in an external application that will be reflected in Cortex AgentX, and vice versa. If an issue is mirrored with an external application, you have the following options:



- **Link the ticket to the issue:** If an issue is linked to a ticket, the ticket number is displayed in the Overview section of the issue card. You see details about the status of the ticket by clicking on the ticket number.
- **Sync changes between the issue and the ticket:** If an issue is synced to a ticket, changes are synchronized in an outbound, inbound, or bi-directional flow.

#### **NOTE:**

Multiple tickets can be linked to an issue with outbound syncing. Issues with inbound syncing can be linked to a single ticket only.

Set up an external integration to sync with issues

Before you can sync issues with external applications, you must set up and configure your integration instance. Complete the following steps:

1. Install the content pack.
  1. To install from the Data Sources & Integrations page: Navigate to Settings → Data Sources & Integrations, click + Add New., and search for the relevant content pack.
  - To install from Marketplace: Navigate to Settings → Configurations → Marketplace, and browse for the relevant content pack.
2. Install the relevant content pack, for example Atlassian Jira or ServiceNow.
2. Connect an integration instance.
  1. Navigate to Settings → Data Sources & Integrations.
  2. Search for the relevant data source (for example Atlassian Jira) select it, and click Add Instance.
  3. Enter instance details in the required fields and click Connect.

Manually create a synced ticket

#### **PREREQUISITE:**

You must set up an integration before you can sync issues. For more information, see Set up an integration for mirroring issues.

You can manually sync existing issues with external applications.

1. From the **Issues** page, right-click an issue and select Run Automation → Select Automation.
2. Under Quick Actions, select the action you want to configure, such as Create Jira Ticket or Create ServiceNow Ticket.
3. Define the required ticket parameters.

#### **NOTE:**

Using issue fields as variables is not currently supported.

4. Under Using, select the name of the instance to execute the command.

#### **WARNING:**

If you leave this field blank, all configured instances will be used.

5. Under Sync Configuration, the following options are displayed, depending on your selection:

- Link to issue: select this option if you want the issue to be linked to the created ticket. You must check this option if you want to sync the issue with the ticket.
- Sync Direction: select the syncing configuration:
  - Inbound: Sync changes from the external ticket with the Cortex AgentiX issue.
  - Outbound: Sync changes from the Cortex AgentiX issue with the external ticket.
  - Bi-directional: Sync changes in both directions.
  - None: Do not sync changes between the Cortex AgentiX issue with the external ticket. If you select this option, the tickets are still linked, but changes are not synced. You can update this option at any time to start syncing.
- Define the inbound and/or outbound sync profiles.

Depending on the selected option, select sync profiles that define field mapping between the issue and the external ticket. You can use the default sync profiles or you can create custom profiles. For more information about sync profiles, see Create a sync profile.

#### **NOTE:**

You can only define a single inbound profile. If you change the inbound sync profile the current profile is overwritten.

You can define multiple outbound profiles; one issue can update multiple tickets.



6. Click OK.

After ticket creation, the ticket number is shown in the Issue card. Click on the ticket number to see details about the created ticket and syncing configuration. In addition, the execution is recorded in the **War Room** tab. If there is an error in the requested action, you can see details in the audit.

7. View or edit the syncing configuration. For more information, see [View, update, or resolve a ticket](#).

Example 20.

The following example shows an automation run on an issue to create a ServiceNow ticket that is synced in an outbound flow with the ticket.

Select an Automation to run on selected issue 430, "Amazon EC2 instance exposed to the public internet" (i)

SET ACTION PARAMETERS

Create ServiceNow Ticket

Description\* (i)  
EC2 instance 'i-065g8973df9e7vh68' in AWS is exposed to the internet

Severity\* (i)  
1 - High x

Short Description\* (i)  
Instance ID: 'i-065g8973df9e7vh68' exposed to the internet

Ticket Type\* (i)  
incident x

Using  
ServiceNow x

+ Set optional parameters

Sync Configuration

Link to issue (i)

Sync Direction (i)  
Outbound

Outbound Profile (i)  
Default ServiceNow V2 Outbound x

Cancel Ok

Run a War Room command to create and sync a ticket

You can run the following command in the War Room to create an external ticket and define the syncing configuration:

```
!jira-create-issue-quick-action summary=<summary> project_key=<key> issue_type_name=<type>
description=<description> using=<instance> mirroring_link_to_object=<true>
mirroring_sync_direction=<syncDirection> mirroring_outbound_profile_id=<profileID>
```

**TIP:**

You can find a sync profile ID under **Settings** → **Configurations** → **Object Setup** → **Issues** → **Sync Profiles**. By default the ID field is not displayed in the table. Click the three dot menu and add it to the table layout.

Example 21.

The following example creates a Jira Bug ticket for the Project Key SCRUM, with an Outbound sync configuration:

```
!jira-create-issue-quick-action summary="Restrict ingress on AWS Network ACLs for admin ports 22 and 3349"
project_key="SCRUM" issue_type_name="Bug" description="We identified that multiple AWS Network ACLS are
allowing inbound (ingress) traffic on admin ports" using="JiraV3" mirroring_link_to_object="true"
mirroring_sync_direction="OUTBOUND" mirroring_outbound_profile_id="h8e14996-8695-5396-9g87-f08suo907486"
```

Create an automation rule for syncing issues with external tickets

**PREREQUISITE:**

You must set up an integration before you can sync issues. For more information, see [Set up an integration for mirroring issues](#).



You can set up automation rules that create external tickets when certain issues occur and define the syncing configuration for transferring data between the issues and tickets.

1. Go to Investigation & Response â†’ Automation â†’ Automation Rules.
2. Click Add Automation Rule.
3. Enter a name and description for the rule.
4. Select whether to enable the rule after creation.
5. Under Rule Conditions, define the WHEN, and IF conditions. For more information about rule conditions, see Create an automation rule.
6. Under THEN select the desired automation, such as Create Jira Ticket and complete the following fields:

1. Define the required ticket parameters.

**NOTE:**

Using issue fields as variables is not currently supported.

2. Under Using, select the name of the instance to execute the command.

**WARNING:**

If you leave this field blank, all configured instances will be used.

3. Under Sync Configuration, the following options are displayed, depending on your selection:

- Link to issue: select this option if you want the issue to be linked to the created ticket. You must check this option if you want to sync the issue with the ticket.
- Sync Direction: select the syncing configuration:
  - Inbound: Sync changes from the external ticket with the Cortex AgentiX issue.
  - Outbound: Sync changes from the Cortex AgentiX issue with the external ticket.
  - Bi-directional: Sync changes in both directions.
- Define the inbound and/or outbound sync profiles.

Depending on the selected option, select sync profiles that define field mapping between the issue and the external ticket. You can use the default sync profiles or you can create custom profiles. For more information about sync profiles, see Create a sync profile.

**NOTE:**

You can only define a single inbound profile. If you change the inbound sync profile the current profile is overwritten.

You can define multiple outbound profiles; one issue can update multiple tickets.

4. Click OK.

If a ticket is created, the ticket number is shown in the Issue card. You can click on the ticket number to see details about the created ticket and syncing configuration. In addition, the execution is recorded in the War Room tab. If there is an error in the requested action, you can see details in the audit.

7. Click Create.

The rule is added to the Automation Rules page. If required, drag to reorder the rules.

Example 22.

The following example shows an automation rule that creates a Jira ticket with bi-directional syncing when a Critical Posture issue is triggered.



## Create New Automation Rule

\* Rule Name  
Create Jira ticket for Critical Posture issues

Description  
Create a Jira ticket with bi-directional syncing for Critical Posture issues

Status  Enabled

\* Rule Conditions

WHEN the following trigger occurs Issue is created

IF the following conditions match  
(issue domain = Posture AND severity = Critical) [Edit](#)

THEN perform the following action

[Create Jira Ticket](#) [Change](#)

### View, update, or resolve a ticket

Once you have set up ticket syncing, you can view, update and resolve the issue and external ticket as required. The changes are reflected according to the defined syncing configuration.

1. To open the ticket details, in the Overview section of the issue card, click on the external ticket number.

A panel opens with details of the external ticket. You can see the external ticket number, the sync configuration, and details of the ticket.

2. Open the linked ticket by clicking on the external ticket number in the panel.

3. Update the fields as required.

The updates are logged in the ticket history.

#### NOTE:

- The inbound syncing flow runs every two minutes, and the outbound syncing flow runs every five minutes.
- In a bi-directional set-up, if the same field is updated in both tickets, the most recently updated value is used.
- In the external ticket, the logged history shows updates to the ticket. The user name that is logged with the history reflects the user token of the user who configured the data source.

4. Resolve the ticket.

#### NOTE:

After an issue is resolved, ticket syncing remains active for up-to seven days. Therefore, you still update, change, or reopen the issue or external ticket and the tickets will continue to sync.

### Edit or disable ticket syncing

You can change the syncing configuration between a ticket and an issue from the issue card.

1. In the Overview section of the issue card, click on the external ticket number.

A panel opens with details of the ticket.

2. Click on the settings icon.

3. Under Sync Configuration, change the syncing configuration as required.

#### NOTE:

If you change the selected inbound sync profile, the original sync profile is immediately overwritten.

4. To disable ticket syncing, take one of the following actions:



- To pause ticket syncing, set the Sync Direction value to None.

This temporarily stops the tickets from syncing, but the tickets are still linked. You can update the syncing configuration at any time to resume ticket syncing.

- To unlink the tickets, uncheck Link to issue.

This action is not reversible.

## 5. Click Save.

Add playbook tasks to create external tickets

### **PREREQUISITE:**

You must set up an integration before you can sync issues. For more information, see [Set up an integration for mirroring issues](#).

You can add a playbook task that creates external tickets and defines the syncing configuration.

1. Open a new or existing playbook and add a new task.

2. Select the Task Type and add a task name.

3. Select one of the following scripts:

- `jira-create-issue-quick-action (Jira V3)`
- `servicenow-create-issue-quick-action (Jira V3)`

4. Under Inputs, add fields for the ticket parameters.

Example 23.

This example defines fields for a Jira ticket.

- Summary: AWS Network ACLs allow ingress traffic on Admin ports
- Project Key: SCRUM
- Issue Type: Bug
- Description: We identified that multiple AWS Network ACLS are allowing inbound (ingress) traffic on admin ports

## 5. Under Sync Configuration, the following options are displayed, depending on your selection:

- Link to issue: select this option if you want the issue to be linked to the created ticket.
- Sync Direction: select the syncing configuration:
  - Inbound: Sync changes from the external ticket with the Cortex AgentiX issue.
  - Outbound: Sync changes from the Cortex AgentiX issue with the external ticket.
  - Bi-directional: Sync changes in both directions.
  - None: Do not sync changes between the Cortex AgentiX issue with the external ticket.
- Define the inbound and outbound sync profiles.

Depending on the selected option, select sync profiles that define field mapping between the issue and the external ticket. You can use the default sync profiles or you can create custom profiles. For more information about sync profiles, see [Create a sync profile](#).

### **NOTE:**

You can only define a single inbound profile. If you change the inbound sync profile the current profile is overwritten.

You can define multiple outbound profiles; one issue can update multiple tickets.

## 6. Save the playbook.

**Limitations of issue mirroring**

Consider the following limitations of issue mirroring:



- Issue syncing requires the latest version of Atlassian Jira (V3) and ServiceNow (V2).
- Issue syncing is currently supported in Atlassian Jira (V3) and ServiceNow (V2) only.
- You can sync up to 50K objects.
- You can create a maximum of 200 sync profiles.
- Cortex AgentiX supports up-to 100 Inbound syncs across all synced tickets over a two-minute time period. Any additional changes beyond this limit will not be synced.
- If a connector instance is deleted or disabled, tickets are no longer synced and external ticket information is not available.
- Custom statuses are not supported.
- Currently, a specific set of fields is supported.

### 3.7.5.1 | Create a sync profile

#### Abstract

You can set up inbound and outbound sync profiles to define field mapping between Cortex AgentiX issues and an external application.

Sync profiles provide a blueprint for how information is exchanged between Cortex AgentiX issues and external applications, by defining field mapping. This ensures that relevant data, such as Status or Description, is accurately transferred and maintains consistency, even if the systems use different terminology.

When you link an issue with an external application (such as Jira), or set up an automation, you can select the sync profile you want to use. Cortex AgentiX provides default outbound and inbound sync profiles, or you can create custom sync profiles as described in the following procedure.

#### How to create a sync profile

1. Go to Settings → Configurations → Object Setup → Issues → Sync Profiles.
2. Click New Profile.
3. Type a profile name and description.
4. Under Integration, select the external application with which you want to map fields, such as Jira V3 or ServiceNow V2.
5. Under Sync Direction, select Inbound or Outbound.

If you select Inbound, you will define field mapping from the external application to Cortex AgentiX. If you select Outbound, you will define field mapping from Cortex AgentiX to the external application.

#### **NOTE:**

If an issue is using bi-directional syncing, you need to provide both an Inbound and an outbound sync profile.

6. Under Field Mapping, select a field to map and select the corresponding field. For example, Jira: Priority, Cortex: Severity.
7. Define one or more values for each field that you want to map.

#### **NOTE:**

- Blank fields are skipped.
- You must define exact values.
- Custom status values are not currently supported.
- Support is currently limited to a specific set of fields.

8. Click Save.

Example 24.

In this example, the sync profile specifies Inbound mapping from Jira v3 fields to Cortex fields.



## New Sync Profile

\* Sync Template Name  
Jira priority and status map

Description  
Inbound Jira priority and status fields

\* Integration  
Jira V3

\* Sync Direction  
Inbound

**Field Mapping**

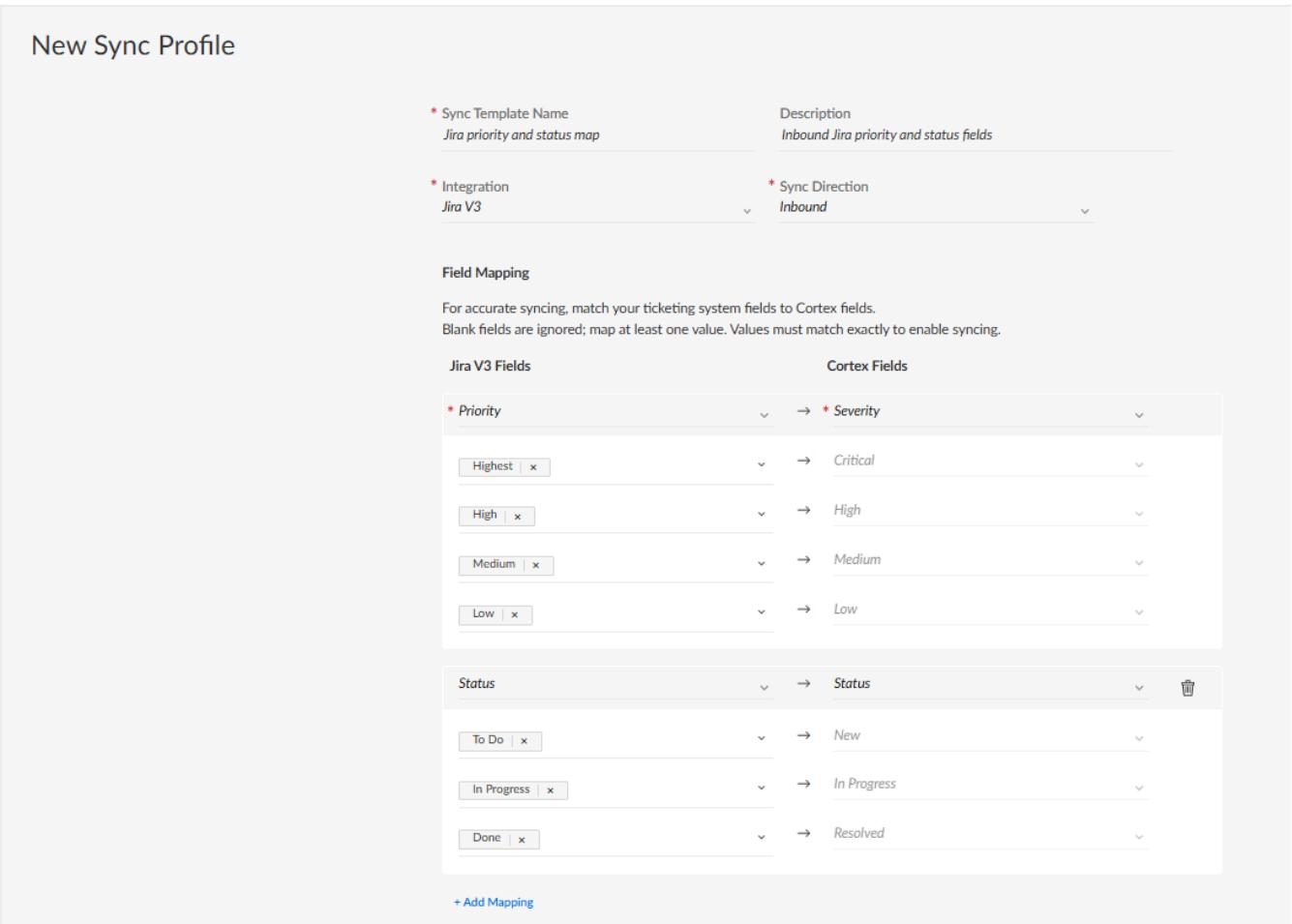
For accurate syncing, match your ticketing system fields to Cortex fields.  
Blank fields are ignored; map at least one value. Values must match exactly to enable syncing.

Jira V3 Fields	Cortex Fields
* Priority	→ * Severity
Highest   x	→ Critical
High   x	→ High
Medium   x	→ Medium
Low   x	→ Low

Status	→	Status
To Do   x	→	New
In Progress   x	→	In Progress
Done   x	→	Resolved

+ Add Mapping



### 3.7.6 | What's a correlation rule?

#### Abstract

Correlation rules help you analyze correlations of multi-events from multiple sources by using the Cortex Query Language based engine for creating scheduled rules.

Correlation rules help you analyze correlations of multiple events from multiple sources by using the Cortex Query Language (XQL) based engine for creating scheduled rules. Issues are then generated based on these correlation rules with a defined time frame and set schedule, including every X minutes, once a day, once a week, or a custom time.

Some examples of events for which you might want to create correlation rules are:

- A user has a number of failed logins, and then a successful login within a small window.
- A device on a watch list has an activity.
- A device connects to an IP that's on a watch list.
- Two specific events occur in a 10 minute window.

After you configure your correlation rules, you can manage them in Threat Management → Detection Rules → Correlations, and view and analyze the generated issues in Cases and the Issues Table. In addition, issues generated by correlation rules are factored into the number of cases displayed in the dashboards.

#### 3.7.6.1 | Correlation rule details

#### Abstract

In the Correlation Rules page, you can view all of your enabled rules in a table format and the various fields displayed.

If you are assigned a role that enables Investigation → Rules privileges, you can manage all user-defined Correlation Rules from Threat Management → Detection Rules → Correlations.

By default, the Correlation Rules page displays all enabled rules. To search for a specific rule, use the filters above the results table to narrow the results. From the Correlation Rules page, you can manage existing rules using the right-click pivot menu. You can also import and export rules in JSON format, which can



help you to transfer your configurations between environments for onboarding, migration, backup, and sharing. You can bulk export and import multiple rules at a time.

In addition, the Correlation Rules page enables you to easily identify and resolve correlation rules errors. The number of errors is indicated at the top of the page in red using the format <number> errors found. You can change the view to only display the correlation rules with errors by selecting Show Errors Only. The LAST EXECUTION column in the table indicates a correlation rule with an error by displaying the last execution time in a red font and providing a description of the correlation rule error when hovering over the field. The following error messages are displayed in the applicable scenarios.

- Invalid query
- Query timeout
- Dependency correlation did not complete
- Unknown error
- Delayed rule— This rule is running past its scheduled time, which can cause delayed results.
- Dataset does not exist: <name of dataset>

**NOTE:**

Only an administrator can create and view queries built with an unknown dataset that currently does not exist in Cortex AgentiX .

A notification is also displayed in Cortex AgentiX to indicate these correlation rules errors.

**NOTE:**

For more information on troubleshooting server errors in scheduled correlation rules, Troubleshoot server errors in scheduled correlation rules.

Correlation rule fields in alphabetical order

**NOTE:**

Certain fields are exposed and hidden by default. An asterisk (\*) is beside every field that is exposed by default.

Field	Description
# OF ISSUES*	The number of issues generated by this rule.
ALERT CATEGORY*	Type of issue as configured when creating the rule. <ul style="list-style-type: none"><li>• Collection</li><li>• Credential Access</li><li>• Dropper</li><li>• Evasion</li><li>• Execution</li><li>• Evasive</li><li>• Exfiltration</li><li>• File Privilege Manipulation</li><li>• File Type Obfuscation</li><li>• Infiltration</li><li>• Lateral Movement</li><li>• Persistence</li><li>• Privilege Escalation</li><li>• Reconnaissance</li><li>• Tampering</li><li>• Other</li></ul>



Field	Description
DATASET*	<p>The text displayed here depends on the resulting action configured for the correlation rule when the rule was created.</p> <ul style="list-style-type: none"> <li>• alerts— When your resulting action for the rule was configured to Generate issue.</li> <li>• Dataset name— When your resulting action for the rule was configured to Save to dataset.</li> </ul>
DESCRIPTION*	<p>The description for the Correlation Rule that was configured when the rule was created.</p>
DRILL-DOWN QUERY	<p>Displays the Drill-Down Query that you configured for additional information about the issue for further investigation using Cortex Query Language (XQL) when you created the rule. If you did not configure one, the field is left empty.</p> <p>After configuration, any issue generated for the Correlation Rule has a right-click pivot menu Open Driltdown Query option, an Open driltdown query link after you investigate any contributing events, and a quick action Open Driltdown Query icon (🔗) that is accessible in the Issues page, which opens a new browser tab in XQL Search to run this query. If you do not define a Drill-Down Query, no right-click menu option, link, or icon is displayed.</p> <p>The Drill-Down Query Time Frame can be configured as either.</p> <ul style="list-style-type: none"> <li>• Generated Issue— Uses the time frame of the issue that is generated, which is the first event and last event timestamps for the issue (default option).</li> <li>• XQL Search— Uses the time frame from when the Correlation Rule was run in XQL Search.</li> </ul>
FAILURE REASON	<p>For a Correlation Rule with an error, displays the error message, which can be one of the following.</p> <ul style="list-style-type: none"> <li>• Invalid query</li> <li>• Query timeout</li> <li>• Dependency correlation did not complete</li> <li>• Unknown error</li> <li>• Delayed rule— This rule is running past its scheduled time, which can cause delayed results.</li> <li>• Dataset does not exist: &lt;name of dataset&gt;</li> </ul> <p><b>NOTE:</b></p> <p>Only an administrator can create and view queries built with an unknown dataset that currently does not exist in Cortex AgentiX.</p>
INSERTION DATE	<p>Date and time when the Correlation Rule was created.</p>
LAST EXECUTION*	<p>Date and time when the correlation rule was last executed. Indicates a correlation rule with an error by displaying the last execution time in a red font and providing a description of the correlation rule Error when hovering over the field. Indicates Real-time in the case of Real Time Correlation Rules.</p>
MITRE ATT&CK TACTIC*	<p>Displays the type of MITRE ATT&amp;CK tactic the correlation rule is attempting to trigger.</p>



Field	Description
MITRE ATT&CK TECHNIQUE*	Displays the type of MITRE ATT&CK technique and sub-technique the correlation rule is attempting to trigger.
MODIFICATION DATE*	Date and time when the correlation rule was last modified.
NAME*	Unique name that describes the rule.
RULE ID	Unique identification number for the rule.
SCHEDULE*	<p>Displays the Time Schedule for the frequency of running the XQL Search definition set for the correlation rule when the rule was created. The options displayed are one of the following.</p> <ul style="list-style-type: none"> <li>• Every 10 Minutes</li> <li>• Every 20 Minutes</li> <li>• Every 30 Minutes</li> <li>• Hourly</li> <li>• Daily</li> <li>• Displays the Time Schedule as Cron Expression fields.</li> </ul>
SEVERITY*	<p>Correlation rule severity that was defined when the correlation rule was created. Severity levels can be Informational, Low, Medium, High, Critical, and Customized.</p> <p>If a generated issue has severity Medium or above, a case is automatically opened. Low severity issues generated by correlation rules are not grouped into cases.</p>
SOURCE*	User who created this correlation rule.
STATUS	Rule status: Enabled or Disabled.
SUPPRESSION DURATION*	The duration time for how long to ignore other events that match the issue suppression criteria that was configured when the rule was created. This is required to configure.
SUPPRESSION FIELDS*	The fields that the issue suppression is based on, which was configured when the rule was created. The fields listed are based on the XQL query result set for the rule. This is optional to configure.
SUPPRESSION STATUS*	Displays the Suppression Status as either Enabled or Disabled as configured when the rule was created.
TIME FRAME*	Displays the time frame for running a query, which can be up to 7 days as configured when the rule was created.



Field	Description
TIMEZONE	Displays the Timezone when the Time Schedule for the frequency of running the XQL Search definition set for the correlation rule is set to run daily or using a cron expression. Otherwise, this field is left empty.
XQL SEARCH	Displays the XQL definition for the correlation rule that was configured in XQL Search when the rule was created.

### 3.7.6.2 | Create a correlation rule

#### Abstract

Create new correlation rules from either the Correlation Rules page or when building a query in XQL Search, or import a many correlation rules from a file.

#### PREREQUISITE:

To enable pivots from issues to a third-party source system, ensure that you know the dataset field name that contains the URL of the source system. Some vendors already provide this URL as part of their API, and you can find it in the third-party product dataset. If there is no URL, you cannot enable this feature.

You can create a new correlation rule from either the Threat Management â Detection Rules â Correlation Rules page or when building a query in XQL Search. You can also import a number of correlation rules.

When setting up correlation rules, you have the following capabilities:

- Specify whether the correlation rule is Scheduled, or scans the data in Real Time, as itâs ingested.
- Define when the correlation rule runs.
- Define whether issues generated by the correlation rule are suppressed by a duration time and a field.
- Set the resulting action for the correlation rule, which includes any of the following:
  - Generate an issue: You can also define the issue settings, which include the Issues Field Mapping for incident enrichment, Issue domain, Issue Severity, MITRE Attack Tactics and Techniques, and other issue settings.
  - Save data to a dataset: Use this option to test and fine-tune new rules before initiating issues and applying correlation use cases.
  - Add data to a lookup dataset
  - Remove data from a lookup dataset

#### NOTE:

- When creating a Real Time Correlation Rule, you can only generate an issue as the resulting action for the Correlation Rule. All other options are disabled.
- To ensure your Correlation rules generate issues efficiently and do not overcrowd your Issues table, Cortex AgentiX automatically disables Correlation rules that reach 5000 or more hits over a 24-hour period.

Create a correlation rule from scratch

1. Open the New Correlation Rule editor.

You can do this in two ways:

- From the Correlation Rules page.
  1. Select Threat Management â Detection Rules â Correlations.
  2. Select +Add Correlation.
- From XQL Search.
  1. Select Investigation & Response â Search â Query Builder â XQL Search.
  2. In the XQL query field, define the parameters for your Correlation Rule.
  3. Select Save as â Correlation Rule.

The New Correlation Rule editor is displayed where the XQL Search section is populated with the query you already set in the XQL query field.



2. Configure the General settings.

- Specify a descriptive Name to identify the correlation rule.
- (Optional) Specify a Description for the correlation rule.

3. Use XQL to define the correlation rule in XQL Search field.

Define the correlation rule in the XQL Search field. After writing at least one line in XQL, you can Open full query mode to display the query in XQL Search. You can Test the XQL definition for the rule whenever you want.

**TIP:**

When creating your correlation rule, you can use predefined values for different fields in the editor, such as Alert Name, Alert Description, and Drill-Down Query. For more information, see [Field replacement syntax in correlation rules](#).

**NOTE:**

- When you open the New Correlation Rule editor from XQL Search, this XQL Search field is already populated with the XQL query that you defined.
- An administrator can create and view queries built with an unknown dataset that currently does not exist in Cortex AgentiX . All other users can only create and view queries built with an existing dataset.

When you finish writing the XQL for the Correlation Rule definition, select Continue editing rule to bring you back to the New Correlation Rule editor, and the complete query you set is added to the XQL Search field.

**NOTE:**

- The XQL features for `call`, `top`, and wildcards in datasets (`dataset in (<dataset prefix>_*)`) are currently not supported in Correlation Rules. If you add them to the XQL definition, you will not be able to Create or Save the Correlation Rule.
- The XQL features for `transaction` in datasets (`dataset in (<dataset prefix>_*)`) are currently not supported in Real Time correlation rules.
- The XQL `tag` stage is currently not supported in correlation rules.
- Using the `current_time()` function in your XQL query for a correlation rule can yield unexpected results when there are lags or during downtime. This happens if the correlation rule doesn't run exactly at the time of the data inside the timeframe, for example, when a rule is dependent on another rule, or when a rule is stuck due to an error, and then runs in recovery mode. Instead, we recommend using the `time_frame_end()` function, which returns the timestamp at the end of the time frame in which the rule is executed.

4. Select to run the Correlation Rule in Real Time or Scheduled.

- Real Time Correlation Rules scan the data as it's ingested.
- You can run Real Time Correlation Rules on Cortex AgentiX issues, Cloud audit logs, third party datasets, and Data Models.
- When you start typing a Correlation Rule, Cortex AgentiX can detect that the query can be run in Real Time and recommend that you select Real Time.
- Real Time Correlation Rules only support the following XQL stages: `dataset`, `datamodel`, `filter`, `alter`, `fields`, and `config case_sensitive`. A Real Time Correlation Rule must include an XQL `filter` stage. For more information on these XQL stages, see the Cortex AgentiX XQL Language Reference Guide.
- The following XQL functions are not supported in a Real Time Correlation Rule: `json_extract_scalar_array`, `parse_epoch`, and `time_frame_end`.
- Dataset views are not supported in Real Time Correlation Rules.

5. If the Correlation Rule is Scheduled, configure the Timing settings.



- Time Schedule: Select the Time Schedule for the frequency of running the XQL Search definition set for the Correlation Rule as one of the following.
  - Every 10 Minutes: Runs every rounded 10 minutes at preset 10-minute intervals from the beginning of the hour, such as 10:10 AM, 10:20 AM, and 10:30 AM.
  - Every 20 Minutes: Runs every rounded 20 minutes at preset 20-minute intervals from the beginning of the hour, such as 10:20 AM, 10:40 AM, and 11:00 AM.
  - Every 30 Minutes: Runs every rounded 30 minutes at preset 30-minute intervals from the beginning of the hour, such as 10:30 AM, 11:00 AM, and 11:30 AM.
  - Hourly: Runs at the beginning of the hour, such as 1:00 AM or 2:00 AM.
  - Daily: Runs at midnight, where you can set a particular Timezone.
  - Custom: Displays the Time Schedule as Cron Expression fields, where you can set the cron expression in each time field to define the schedule frequency for running the XQL Search. The minimum query frequency is every 10 minutes and is already configured. You can also set a particular Timezone.

By default, the query is set to run once an hour (1 Hour/s).

- Timezone (Optional): You can only set the Timezone when the Time Schedule is set to Daily or Custom. Otherwise, the option is disabled.
- Query time frame: Set the time frame for running a query, which can be up to 7 days. Specify a number in the field and in the other field select either Minute/s, Hour/s, or Day/s.

#### 6. (Optional) Configure Issue Suppression settings.

Define whether the issues generated by the Correlation Rule are suppressed by a duration time, field, or both.

- Enable issues suppression: Select this checkbox to Enable issue suppression. By default, this checkbox is clear and the issues of the Correlation Rule are configured to not be suppressed.
- Duration time: Set the Duration time for how long to ignore other events that match the issue suppression criteria, which are based on the Fields listed. Specify a number in the field and in the other field select either Minute/s, Hour/s, or Day/s. By default, the generated issues are configured to be suppressed by 1 hour (1 Hour/s). The Duration time can be configured for a maximum of 1 day.
- Fields (Optional): Select the fields that the issue suppression is based on. The fields listed are based on the XQL query result set. You can perform the following.
  - Select multiple fields from the list.
  - Select all to configure all the fields for suppression. This means that all the fields must match for the issues to be suppressed. This option will generate multiple issues during the suppression period.
  - Search for a particular field, which narrows the available options as you begin typing.
  - Do not set any Fields by leaving the field empty only 1 issue is generated during the suppression period.

#### 7. Configure the resulting Action for the Correlation Rule.

You can select one of the following resulting actions to occur, where the configuration settings change depending on your selection:

Generate issue

Generates a Correlation type of issue according to the configured settings in the New Correlation Rule editor (default). When this option is selected, a number of new sections are opened to configure the issue.

Issue Settings



- Issue Name: Specify a name. You can incorporate a variable based on a query output field in the format `$fieldName`.
- Issue Domain: Select the domain to which you want to assign any generated issues.

**NOTE:**

- The incident that contains the issues will also be assigned to the selected domain. For more information, see Case and issue domains.
- If you want to generate a health issue, choose the Health domain. For more information about health issues, see About health issues.

- Severity: Select the severity type whenever an issue is generated for this Correlation Rule as one of the following:

- Informational
- Low
- Medium
- High
- Critical
- User Defined: Select fields from inside the query.

**NOTE:**

- If a generated issue has severity Medium or above, a case is automatically opened.
- Low severity issues generated by correlation rules are not grouped into cases.

- Category: Select the type of issue that is generated. The list of categories reflects the selected domain.
- Issue Description (Optional): Specify a description of the behavior that will raise the issue. You can include dollar signs (\$), which represent the fields names (i.e. output columns) in XQL Search.

For example.

```
The user $user_name has made $count failed login requests to $dest in a 24 hours period
```

Output.

```
The user lab_admin has made 234 failed login requests to 10.10.32.44 in a 24 hours period
```

**NOTE:**

There is no validation or auto complete for these parameters and the values can be null or empty. In these scenarios, Cortex AgentiX does not display the null or empty values, but adds the text **NULL** or **EMPTY** in the descriptions.

- Drill-Down Query (Optional): You can configure a Drill-Down Query for additional information about the issue for further investigation using XQL. This XQL query can accept parameters from the issue output for the Correlation Rule. Yet, keep in mind that when you create the Correlation Rule, Cortex AgentiX does not know in advance if the parameters exist or contain the correct values. As a result, Cortex AgentiX enables you to save the query, but the query can fail when you try and run it. You can also refer to field names using dollar signs (\$) as explained in the Issue Description.

Once configured any issue generated for the Correlation Rule has a right-click pivot menu Open Drilldown Query option, an Open drilldown query link after you investigate a contributing event, and a quick action Open Drilldown Query icon (🔗) that is accessible in the Issues page, which opens a new browser tab in XQL Search to run this query. If you do not define a Drill-Down Query, no right-click pivot menu option, link, or icon is displayed.

- Drill-Down Query Time Frame: Select the time frame used to run the Drill-Down Query from one of the following options, which provides more informative details about the issue generated by the Correlation Rule.

- Generated Issue: Uses the time frame of the issue that is generated, which is the first event and last event timestamps for the issue (default option). If there is only one event, the event timestamp is the time frame used for the query.

- XQL Search: Uses the time frame from when the Correlation Rule was run in XQL Search.

- MITRE ATT&CK (Optional): Select the MITRE Tactics and MITRE Techniques you want to associate with the issue using the MITRE ATT&CK matrix.

- You can access the matrix by selecting the MITRE ATT&CK bar or Open complete MITRE matrix link underneath the bar on the right.

- Select the MITRE Tactics listed in the first row of the matrix and the applicable MITRE techniques and Sub-Techniques, which are listed in the other rows in the table. You can select either MITRE Tactics only, MITRE techniques and Sub-Techniques only, or a combination of both.

- Click Select and the matrix window closes and the MITRE ATT&CK section in the New Correlation Rule editor lists the number of Tactics and Techniques configured, which is also listed in the bar. For example, in the following image, there are 3 Tactics and 4 Techniques configured. The three MITRE Tactics are Resource Development with 2 Techniques configured, Credential Access with 1 Technique configured, and Discovery with 1 Technique configured.

## Issues Fields Mappings

You can map the issue fields to display the mapped fields in the Issues page to provide important information in analyzing your issues. In addition, mapping the fields helps to improve incident grouping logic and enables Cortex AgentiX to list the artifacts and assets based on the map fields in the



incident. The options available can change depending on your Correlation Rule definitions in XQL Search. Each preconfigured field that is automatically mapped is clearly displayed. There are two ways to map the issue fields.



- Use the preconfigured Cortex AgentiXIssue field mapping

Select this option if you want Cortex AgentiX to automatically map the fields for you. This checkbox only displays when your Correlation Rule can be configured to use Cortex AgentiX incident enrichment, and then it is set as the default option. We recommend using this option whenever it is available to you.

When creating a Correlation Rule that runs on the Cortex Data Model (XDM), we recommend that this checkbox is selected. Cortex AgentiX automatically maps the XDM fields to issue fields as listed in the table below. This means that these XDM field values are displayed in the Issues page.

**NOTE:**

When more than one XDM field is listed, Cortex AgentiX looks for the field value according to the order of the fields listed.

You can edit the source mapped to preconfigured fields to customize the mapping to your needs. Click the Edit icon next to a field value to select one of the other values.

Manually map the issue fields by selecting the fields that you want to map. When you create the Correlation Rule, Cortex AgentiX does not detect whether the issue fields that you mapped manually are valid. If the fields are invalid according to your mapping, null values are assigned to those fields.

**NOTE:**

When more than one XDM field is listed, Cortex AgentiX looks for the field value according to the order of the fields listed.

XDM fields mapped to issue fields

XDM Fields	Cortex AgentiX Issue Fields
xdm.source.host.fqdn	Domain
xdm.event.type	Event Type
xdm.target.file.filename, xdm.target.module.filename, xdm.email.attachment.filename	File name
xdm.target.file.sha256, xdm.target.module.sha256, xdm.email.attachment.sha256	File SHA256
xdm.source.host.ipv4_addresses	Host IP
xdm.source.host.hostname	Host Name
xdm.source.host.os_family	Host OS
xdm.source.process.executable.filename, xdm.source.process.name	Initiated By
xdm.source.process.command_line	Initiator CMD
xdm.source.process.executable.path	Initiator path
xdm.source.process.executable.sha256	Initiator SHA256
xdm.source.process.executable.signature_status	Initiator signature
xdm.source.process.executable.signer	Initiator signer



XDM Fields	Cortex AgentiX Issue Fields
xdm.source.ipv4, xdm.source.host.ipv4_addresses	Local IP
xdm.target.process.executable.signature_status	Process execution signature
xdm.target.process.executable.signer	Process execution signer
xdm.target.host.hostname	Remote Host
xdm.target.ipv4, xdm.target.host.ipv4_addresses	Remote IP
xdm.target.port	Remote port
xdm.target.process.command_line	Target process CMD
xdm.target.process.executable.filename, xdm.target.process.name	Target Process Name
xdm.target.process.executable.path	Target Process Path
xdm.target.process.executable.sha256	Target process SHA256
xdm.target.url	URL
xdm.source.user.username	User name

#### User Defined Fields

You can add more fields to the current preconfigured listing. To do so, under User Defined Fields, click Add field and select from the options. When Cortex AgentiX adds a new preconfigured destination field, if you have already defined this destination field, your selection overrides the out-of-the-box mapping to that destination.

- Manually map the issue fields by selecting the fields that you want to map. When you create the Correlation Rule, Cortex AgentiX does not detect whether the issue fields that you mapped manually are valid. If the fields are invalid according to your mapping, null values are assigned to those fields.

#### **NOTE:**

When Use the Cortex AgentiX default incident enrichment is not selected and you have not mapped any issue fields, the issue is dispatched into a new incident.

(Optional) Set fields as default for new Security Domain correlations - saves the custom issue fields that are configured for this rule for all users. When a user next creates an incident for the same domain, these fields are automatically configured instead of the default field set.

To restore the system defaults, click Restore default system fields.

- (Optional) To enable pivots from issues to a third-party source system, in the Issues Fields Mapping section, click +Add field and select the External URL option. For this field, search for and select the dataset field which stores the third-party URL.

#### Save to dataset

Use to save the data generated from the Correlation Rule to a separate Target Dataset. This option is helpful when you are fine-tuning and testing a rule before promoting the rule to production. You can also save a rule to a dataset as a building block for the next Correlation Rule, which will be based on the results of the first Correlation Rule instead of building too complex XQL queries.

You can either create a new Target Dataset by specifying the name for the dataset in the field or select a preexisting Target Dataset that was created for a different Correlation Rule. The list only displays the datasets configured when creating a Correlation Rule. Different Correlation Rules can be saved to the same dataset and Cortex AgentiX will expand the dataset schema as needed. The dataset you configure for the Correlation Rule contains the following additional fields:



- `_rule_id`
- `_rule_name`
- `_insert_time`

Add to lookup

Use to add data to a specified lookup dataset. After selecting this option, perform the following:

1. In the Target Dataset field, select an existing lookup dataset to add the data.

In the displayed mapping, Cortex AgentiX lists fields from the lookup schema in the KEY column to enable you to map fields from the query to an entry in the lookup.

2. In the VALUE column, map at least one field from the query to an entry in the lookup dataset (KEY column).

3. (optional) You can set a single field or multiple fields as unique by selecting the checkbox in the UNIQUE column. A unique field means these fields are designated as a key to update existing entries as opposed to creating a new entry. If multiple fields are selected, these fields together are used to identify existing entries. If several existing entries meet the condition, all these entries are updated. If no existing entries meet the condition, the entry is added as a new one. If no field is marked as unique, records are added as new.

#### **IMPORTANT:**

The maximum size of a lookup dataset is 50 MB. If the data exceeds this limit, the add to lookup action fails.

Remove from lookup

Removes data from a specified lookup dataset. After selecting this option, perform the following:

1. In the Target Dataset field, select an existing lookup dataset to remove data.

In the displayed mapping, Cortex AgentiX lists fields from the lookup schema in the KEY column to enable you to map fields from the query to an entry in the lookup.

2. In the VALUE column, map at least one field from the query to an entry in the lookup dataset (KEY column). All rows (lookup entries) matching these field mapping values (filtering condition) will be deleted. If several existing entries meet the condition, all these entries are deleted. If no existing entries meet the condition, no entries are deleted.

8. (Optional) Disable the Correlation Rule.

Select Disable â Create if you want to finish configuring your Correlation Rule at a different time, but do not want to lose your settings. The Create button is only enabled when you have configured all the mandatory fields in the New Correlation Rule editor. Once configured, your Correlation Rule is listed in the Correlation Rules page, but is disabled. You can edit or enable the rule at any time by right-clicking the rule and selecting Edit Rule or Enable.

9. Create the correlation rule.

The rule is added to the table in the Correlation Rules page as an active rule and a notification is displayed.

Import correlation rules from a file

You can import a number of correlation rules from a JSON file. This facilitates the sharing of correlation rules between tenants.

To import a file containing correlation rules, select Threat Management â Detection Rules â Correlations and click Import at the top right corner of the page.

#### 3.7.6.3 | Manage correlation rules

Abstract

View and manage your correlation rules

View and manage your correlation rules in Threat Management â Detection Rules â Correlations. To manage a Correlation Rule, right-click the Correlation Rule and select an action.

You can also monitor your correlation rule executions with the `correlations_auditing` data set. For more information, see Monitor correlation rules.

Right-click actions for managing correlation rules



- View related issues: View the issues generated by this correlation rule in the Issues page. You can Show issues in new tab or Show issues in same tab.
- Open in XQL: View the XQL results for the correlation rule in XQL Search. You can Show results in new tab or Show results in same tab.
- Execute Rule: Run the rule now without waiting for the scheduled time.

**NOTE:**

Execute Rule is not available for real time correlation rules.

- Preview Rule: View the rule before it's executed.
- Save as new: Duplicate the correlation rule and save it as a new correlation rule.
- Export: Select one or more rules to export to a JSON file.
- Disable the selected correlation rule. This option is only available on an active rule.
- Enable the selected correlation rule. This option is only available on an inactive rule.
- Edit Rule: Edit the rule parameters configured in the Edit Correlation Rule editor.
- Delete the correlation rule.
- Copy entire row to copy the text from all the fields in a row of a correlation rule.
- Show rows with <field value> to filter the correlation rules list to only display the correlation rules with a specific field value that you select in the table. On certain fields that are null, this option does not display.
- Hide rows with <Rule Description> : Filter the correlation rules list to hide the correlation rules with a specific field value that you select in the table. On certain fields that are null, this option does not display.

### 3.7.7 | Manage external dynamic lists

#### Abstract

Configure and manage your external dynamic lists in Cortex AgentiX.

An External Dynamic List (EDL) is a hosted text file. In Cortex AgentiX, you can configure an EDL to share a list of Cortex AgentiX indicators with other products in your network, such as a firewall. For example, your Palo Alto Networks firewall can add IP addresses and domain data from the EDL to block or allow lists.

Cortex AgentiX hosts the following external dynamic lists that you can configure and manage:

- IP Addresses EDL
- Domain Names EDL

**NOTE:**

- To configure an EDL, you must have a role that includes EDL permissions, such as Instance Admin or Account Admin.
- Configuring custom certificates or private API Keys in the EDL integration instance is supported only on engines, not on the Cortex AgentiX server.
- For EDL integrations on the server, you must set a username and password. For long-running integrations running on an engine, we strongly recommend setting a username and password, but it is not required. You can set credentials for all EDL integrations or for a specific integration instance.

If you set a username and password in an EDL integration instance, only those credentials are accepted, and the username and password you set in the Cortex AgentiX doesn't work.

You can set up Cortex AgentiX to export internal data to an EDL using an EDL integration installed either on the Cortex AgentiX tenant or on an engine.

**NOTE:**

The legacy external dynamic list PAN-OS integration is deprecated. Configure the EDL integration on the Data Sources & Integrations page (by clicking the Automation & Feed Integration link).

#### Access EDL Data on the Cortex AgentiX tenant

If the EDL integration runs on the Cortex AgentiX tenant, you must set a username and password to allow external access to the data.

1. Navigate to Settings â Configuration â Integrations â External Dynamic List Integration.
2. Enter a username and password for all EDL integration instances. If you do not enter credentials here, you must set them for each integration instance.
3. Navigate to Settings â Data Sources & Integrations.
4. Select the Generic Export Indicators Service integration and click Add Instance.



5. (Optional) If you want credentials specific to this integration instance, enter a Username and Password .

**NOTE:**

If the EDL integration runs on the Cortex AgentiX server, you do not need to enter a Listen Port in the instance settings. The system auto-selects an unused port for the EDL integration when the instance is saved. If you enter a value for Listen Port, it will be overwritten by the port auto-selected by the system

6. Enter an indicator query.

The query updates the EDL list. To view expected results, run `!findIndicators query=<your query>` from the Cortex AgentiX CLI. Field names in your query must match the machine name for each field.

7. Enter the maximum list size.

8. Run the following curl command to access and test the External Dynamic List: `https://ext-<cortex-agentix-address>/xsoar/instance/execute/<instance-name>`

Example 25.

```
curl -v -u user:pass https://ext-mytenant.paloaltonetworks.com/xsoar/instance/execute/edl_instance_01\?q\=type:ip
```

**IMPORTANT:**

The EDL URL must always be prefixed by ext-.

9. Save your changes.

#### Access EDL Data on an Engine

Provide access to internal data via an engine using an endpoint port. We strongly recommend setting a username and password for additional security.

1. Navigate to Settings → Data Sources & Integrations.

2. Search for and select the Generic Export Indicators Service integration, click Add Instance and enter:

- Listen Port: The service to access the EDL runs on this port from within Cortex AgentiX. You need a unique port for each long running integration instance (do not use the same port for multiple instances).
- (Optional) Username and Password: The username and password for the EDL.
- Run on single engine: Select the engine from a drop-down.

3. Enter an indicator query.

The query updates the EDL list. To view expected results, run `!findIndicators query=<your query>` from the Cortex AgentiX CLI. Field names in your query must match the machine name for each field.

4. Enter the maximum list size.

5. Run the following curl command to access and test the External Dynamic List with the engine URL: `http://<engine-address>:<integration listen port>/`

Example 26.

```
curl -v -u user:pass http://<engine_address>:<listen_port>/?n=50
```

6. Save your changes.

## 3.8 | Automations

Automations leverage playbooks to execute predefined workflows, use context data to make informed decisions, and interact with lists to store and retrieve information as needed during the automation process.

### 3.8.1 | Automation in Cortex AgentiX

#### Abstract

Automate response to issues, using playbooks and Quick Actions, triggered automatically by automation rules or manually from an issue.

Automation enables you to improve efficiency and response times by performing actions on one or more issues, either automatically in response to predetermined conditions or manually triggered during your investigation workflow. In Cortex AgentiX, you can use playbooks, scripts, and commands, and Quick Actions to streamline operations, accelerate triage, and boost productivity.

The Automation Insights dashboard provides a high level overview of your automations.



- Playbooks

Playbooks enable you to organize and document security monitoring, orchestration, and response activities. Playbooks are self-contained, fully documented prescriptive procedures that query, analyze, and take action based on the gathered results.

Playbooks are built from regular tasks, quick actions, and sub-playbooks. Playbook tasks can run out-of-the-box or custom scripts and integrations to communicate with third-party systems. You can use out-of-the-box playbooks as is, or customize them according to your requirements. You can also reuse individual playbook tasks as building blocks for new playbooks, saving time and streamlining knowledge retention.

Playbooks can run automatically on issues based on automation rules, run automatically by jobs on a schedule or based on a delta, or can be run manually on one or more issues.

**NOTE:**

You can build end-to-end automation workflows from within the playbook editor, including creating automation rules, configuring integration instances, and creating and editing tasks. For more information, see [Playbooks](#).

- Scripts and commands

Cortex AgentX includes built-in commands, as well as commands and scripts from the core content packs. In addition, when you adopt playbooks, any necessary scripts and integrations for the playbook are automatically downloaded. You can also write your own scripts or edit existing scripts.

Scripts and commands can be used in playbook tasks or run manually from the War Room.

- Quick Actions

Quick actions are single commands that enable you to respond rapidly without requiring complex playbooks.

Quick Actions can be included within playbooks, run automatically on issues based on automation rules, or run manually on one or more issues.

#### Automation rules

Automation rules enable you to run playbooks or Quick Actions automatically on issues, based on preset criteria. Automation rules follow a WHEN / IF / THEN structure. For example, WHEN an issue is created, IF the severity is critical, THEN set the case assignee to a specific analyst. For more information, see [Create an automation rule](#).

#### Manually trigger automation

Playbooks and Quick Actions can also be run on demand. For more information, see [Run an automation on an issue](#).

### 3.8.2 | Quick Actions

Quick Actions are preset single commands that enable you to automate basic tasks such as creating tickets in third-party systems, sending Slack messages, and changing issue severity.

You can create quick actions using the following:

- **Automation rules:** You can create predefined rules to run Quick Actions as issues are created. For more information, see [Create an automation rule](#)
- **Playbooks:** You can use Quick Actions as tasks within playbooks.

When investigating an issue, in the Issues table, you can right-click to Run an Automation on one or more issues. For more information, see [Run an automation on an issue](#).

By default, Quick Actions run using all available integration instances that contain the command. When selecting a Quick Action to run on an issue or to use for an automation rule, you can also choose one specific integration instance.

When you run an automation from the Issues table, in some cases the system provides recommended Quick Actions, based on the context. Quick Actions may also be provided in Recommended Automation Rules.

**NOTE:**

Quick Actions appear as War Room entries, but do not appear in the Work Plan.

### 3.8.3 | Automation Exclusion Center

#### Abstract

Automation exclusion policies prevent commands and scripts from performing remediation on critical assets.

Automation exclusion policies prevent commands and scripts from performing automated remediation actions on critical assets, such as users, IP addresses, and domains. For example, a playbook task might block multiple domains, but mission-critical domains in the policy list would not be blocked.



Automation exclusion policies apply any time a relevant command or script runs, whether in a playbook task, a Quick Action, or the CLI. If you configure a policy to allow overrides, users can manually run the command in the War Room, using the `override-policy` parameter. Any command triggered with the `override-policy` parameter appears in the Management Audit Logs. If you attempt to use the `override-policy` parameter and the policy does not allow overrides, an error entry appears in the War Room.

When an automation exclusion policy prevents a command or script from a remediation action, the exclusion appears in the issue War Room.

When a playbook task contains a command or script that is included in an automation exclusion policy, a Policy tab appears in the task details pane, showing the relevant policy.

To enable an automation exclusion policy, add critical assets to a list. Each policy uses one or more lists to exclude assets from remediation. By default, all policies are enabled, but lists are empty until assets are added to the list.

#### **NOTE:**

By default, all users have read and edit permissions to lists. When creating a list of critical assets, we recommend limiting the read and edit permissions to specific roles.

User Hard Remediation and User Soft Remediation policies can also use asset groups, enabling automatic updates of critical assets without requiring you to edit a list. These remediation policies can contain lists, asset groups, or a combination of lists and asset groups.

Policies can be enabled or disabled, and lists can be edited, but you cannot add or remove policies.

Each policy can include one or more scripts or commands. Commands and scripts only appear if the content is installed. The policy affects only these scripts and commands. Scripts and commands cannot be added, edited, or removed from the policy.

By default, only admin users have access to the Automation Exclusion Center page. You can also provide other roles with View or View/Edit access to the Automation Exclusion Center. When creating or editing a role, the permission can be found under Investigation & Response → Automations.

Policies can be sorted, filtered, and searched using the category, status, policy, exclude, and description columns.

##### **3.8.3.1 | Manage automation exclusion policies**

Abstract

Automation exclusion policies prevent commands and scripts from performing remediation on critical assets. Edit lists of critical assets and enable/disable policies.

Automation exclusion policies prevent commands and scripts from performing automated remediation actions on critical assets, such as users, IP addresses, and domains. For example, a playbook task might block multiple domains, but mission-critical domains in the policy list would not be blocked.

Admin users and all roles with read/write permissions to the Automation Exclusion Center can edit, disable, and enable policies.

1. Go to Settings → Configurations → Automation → Automation Exclusion Center.

2. Right-click on a policy and choose Edit.

3. From the Edit Policy page, you can do the following:

- Enable or disable the policy. Policies are enabled by default.
- Enable or disable policy overrides. If you enable policy overrides, users can manually run the commands and scripts on the excluded critical assets, using the `override-policy` parameter. Use of the `override-policy` parameter is included in the Management Audit Logs.
- Select one or more lists of excluded assets.

Clicking the list icon opens a new browser tab for the Lists page, where you can create and edit lists.

#### **NOTE:**

For the IAM User Hard Remediation and User Soft Remediation policies, we recommend including username, email, and ID for each user you want to exclude. Example: `username1`, `user@example.com`, `userID112`.

Each list can be filtered by conditions, such as Equals, Ends with, and Doesn't include. For example, you can exclude all email addresses with your company's domain using the Ends with filter.

- For IAM User Hard Remediation and User Soft Remediation policies, you can also select asset groups. These policies can include only lists, only asset groups, or a combination of asset groups and lists.
- Under THEN skip execution of the following commands and scripts, click to view the scripts and commands affected by the policy. Commands only appear if they are part of an active integration instance. You cannot edit the list of scripts and commands.

4. Save your changes.

#### **NOTE:**

You can also right click on a policy from the main Automation Exclusion Center page to disable or enable the policy.

If you click on a list name in the Exclude column, that list opens in the Lists page.



### 3.8.4 | Playbooks

Playbooks automate and standardize workflows, ensuring consistent and efficient case response and management.

#### 3.8.4.1 | Playbooks overview

##### Abstract

Cortex AgentIX playbooks enable you to structure and automate many of your security processes. Parse case information, interact with users, and remediate.

Playbooks are a series of tasks that run in a predefined flow to save time and improve the efficiency and results of the investigation and response process. They enable you to automate many security processes, including handling investigations and managing tickets. For example, a playbook task can parse the information in an issue, whether it is an email or a PDF attachment.

##### One-stop playbook development

Before you start building your playbook, go to the Playbooks page and review the Org playbook list, which are playbooks that are currently used in your organization. On the Playbook Catalog page, you can find available out-of-the-box playbooks that are not in use in your organization which you can adopt and use. If an existing playbook does not meet your use case, you can develop a playbook from scratch. Whether editing an existing playbook or creating a new one, you can manage the entire automation development flow in the playbook editor, including creating and editing tasks, configuring automation rules to trigger your playbooks, and setting up all relevant integrations.

##### Task Library

The Task Library in the playbook editor contains the following objects you can add to your playbook. For example, you can create new tasks from scripts, repurpose existing tasks, and use existing playbooks as sub-playbooks.

Playbook tasks display unique logos to more easily identify task type and origin, for example third-party integration commands, built-in scripts and tasks, and tasks requiring manual inputs.

Task Library Object	Action	See More
Quick Actions	Add single commands requiring minimal configuration.	See topic.
Commands & Scripts	Add commands and scripts from integrations that you install and configure instances for as needed.	See topic.
Playbooks	Add sub-playbooks to your playbook from your Org repository or from the Playbooks Catalog.	See topic.
AI Prompt	Add AI prompts with inputs and outputs that run automatically.	See topic.
Manual Tasks	Add tasks from playbooks in your Org repository.	See topic.
Header	Add section headers to organize your playbook.	See topic.
Blank Task	Create a new task from scratch.	See topic.

##### Post-development playbook testing

After developing the playbook (including setting automation rules to trigger the playbook), run the debugger to initially test the playbook.

After verifying the playbook is triggered and runs properly with issues, it is ready to use in production.

You can see which playbook ran for an issue by going to Cases & Issues, selecting Issues and scrolling to the Playbook column. You can view or update the playbook by selecting an issue and clicking the Work Plan tab. Select another playbook to run from the dropdown list.

You can see which playbook ran in a case, if any, by going to Cases & Issues, selecting Cases and looking at the Automation section in the Overview tab for the case. You can view or update the playbook by going to the Issues & Insights tab, selecting an issue, and then clicking the Work Plan tab. In the Work Plan, you can select another playbook to run from the dropdown list.

For more information, see [Investigate cases](#).



### 3.8.4.2 | Playbook development checklist

#### Abstract

Follow the playbook development flow to create playbooks that structure and automate many of your security processes.

The playbook development checklist follows the logical flow for developing a playbook.



We recommend that you review the following steps to successfully implement your playbook.

Step	Details	See More
Step 1. Plan your playbook	During the initial planning stage when designing your use case, start defining the playbook flow. Consider the process you want to automate and the steps and the decisions during the process. These steps and decisions become the playbook tasks.	<a href="#">See topic</a>
Step 2. Build your playbook	Consider whether to use a playbook out-of-the-box, customize an existing playbook, or create a new playbook from scratch. Create playbook tasks, inputs, and outputs. Maintain playbook versioning to keep track of playbook development history.	<a href="#">See topic</a>
Step 3. Customize your playbook	Fine tune your playbook for your needs, including extracting indicators, extending context, and adding issue fields to the system.	<a href="#">See topic</a>
Step 4. Test your playbook	Debug errors in your playbook. Use playbook metadata to troubleshoot playbook performance.	<a href="#">See topic</a>

### 3.8.4.3 | Plan your playbook

#### Abstract

Considerations when planning your playbook.

When defining the workflow of your playbook, consider the following:

- What processes do you need to automate?
- Are there any decisions that require manual intervention?
- Do you require dynamic, intelligent decision-making or complex data interpretation as part of your automation?
- Are there any time-sensitive aspects to the playbook?
- When is the case considered remediated?

#### Example 27. Review the Phishing use case

Review the following workflow for a phishing use case. Also, review the playbooks in the Phishing content pack to see how they work.

- Detection
- Identification
- Analysis
- Remediation



Each of these high-level processes can contain a number of sub-processes that require step-by-step actions, all of which can be automated with either customized or new playbooks.

#### 3.8.4.4 | Manage playbooks

##### Abstract

Navigate the Playbooks page.

The Playbooks page is organized to help easily access and utilize playbooks specific to your use cases. It contains two main sections, key playbook details on the top and a table listing all the playbooks in your Org repository on the bottom.

##### Playbook status

Playbook statuses enable tracking the progress of automation tasks and identifying any issues or delays. If needed, you can then take corrective actions to ensure smooth workflow execution and operational efficiency. The status includes how many playbooks:

- Are in your Org repository
- Are enabled
- Are active
- Are using an automation rule
- Are used as sub-playbooks
- Ran in the past week

##### The Org repository table

The playbooks listed in the Org repository table have been either adopted by or built by your organization. The table shows high level details about the playbooks, including:

- Playbook name
- Description
- Status
- Source
- Enabled and disabled automation rules associated with the playbook
- How many playbooks it serves as a sub-playbook in
- Last updated
- Updated by
- The content pack the playbook is a part of
- Playbook tags

When you right-click a specific playbook, you can choose to open it in the editor, duplicate, disable, download, or remove it.

Playbooks in your Org Playbooks can be triggered to run by automation rules, jobs, or can be manually run on one or more issues.

Playbooks that you adopted are part of content packs. When a playbook is adopted, the content pack for that playbook is downloaded and appears in Marketplace. If you remove a playbook from your Org Playbooks, the content pack remains installed, but the playbook is no longer available for automation rules or manual runs.

##### Playbook Catalog

The Playbook Catalog contains all the playbooks available in Marketplace, organized by cards. You can search for a playbook, and the system also recommends playbooks based on name, tag, or description.

Clicking a card provides a preview of the playbook. If it is relevant for your use case, click Adopt this playbook to bring it into your Org repository and make it available to run.

##### **NOTE:**



- The library by default shows only playbooks that are not adopted. Click the Show Adopted checkbox to show the adopted playbooks, indicated by an Adopted mark.
- The library shows the most updated playbook version. Adopting an older version than shown should be done through Marketplace.
- Adopting a playbook does not make it run. Some content packs include recommended automation rules. When you configure automation rules, you can view the recommendations. See Create an automation rule.

#### 3.8.4.5 | Build your playbook

##### Abstract

Use an out-of-the-box playbook, customize an existing playbook, or create a new playbook based on your organization's needs.

Depending on your use case, you can use or customize a system playbook or develop a new playbook from scratch.

Developing a new playbook from scratch enables a tailored solution for your use case, whereas customizing a system playbook can save time, reduce complexity, and be a more efficient way to meet your organization's specific security and issue response needs.

Follow these steps to build a playbook.

Task	Description	See More
Task 1. Choose from existing playbooks or create your own	Search for an out-of-the-box playbook to use, customize it, or create one based on your use case.	See topic.
Task 2. Configure playbook settings	Define playbook settings, such as playbook triggers, inputs and outputs, and general settings.	See topic.
Task 3. Add objects from the Task Library	The Task Library contains Quick Actions, scripts, AI tasks, sub-playbooks, and tasks that enable you to communicate with end users, set conditions, and store relevant data.	See topic.
Task 4. Add custom playbook features	Customize your playbook, including adding scripts and sub-playbook loops, filtering and transforming data, extracting indicators, extending context, creating issue fields, and polling.	See topic.
Task 5. Test and debug the playbook	Set breakpoints, conditional breakpoints, skip tasks, and input and output overrides in the playbook debugger.	See topic.
Task 6. Manage playbook content	Save versions of your playbook in Cortex AgentiX.	See topic.

#### 3.8.4.5.1 | Task 1. Choose from existing playbooks or create your own

##### Abstract

Use an existing playbook from your Org repository or search for a playbook in the Playbook Catalog. Customize an existing playbook, or create a new playbook based on your use case.

Go to the Investigation & Response → Automation → Playbooks page to find an existing playbook, customize it, or create a playbook.

Find an existing playbook

Playbooks in your Org Repository have already been adopted by your organization and are available to run. The Playbook Catalog contains all available playbooks in Marketplace that you can adopt into your Org Repository. You can preview before adopting.

- View the list of playbooks on the main Playbooks page in the Org Repository table. You can also search for a playbook that exists in the Org Repository by clicking Add Filter.

Use free text in the search box, entering part or all of the playbooks' names or description. You can also search for an exact match of the playbook name by putting quotation marks around the search text. For example, searching for "**Block Account - Generic**" returns the playbook with that name.

Search for more than one exact match by including the logical operator "or" in-between your search texts in quotation marks. For example, searching for "**Block Account - Generic**" or "**NGFW Scan**" returns the two playbooks with those names. Wildcards are not supported in free text search.



**TIP:**

If there are additional relevant playbooks in Marketplace that are not in your Org repository, you can click Explore them now to see them in the Playbook Catalog and choose to adopt.

2. Click Playbook Catalog to browse all available playbooks in Marketplace that you can adopt. Click Playbook Library to go back to the main Playbooks page.

1. Click a playbook card for a preview of the playbook.

2. Click Adopt this playbook to add the playbook to your Org repository.

A confirmation message displays when the playbook is successfully added.

3. Click View in Org Playbooks to select the adopted playbook from the Org repository table.

You can use the playbook as-is, or customize it as needed.

Edit a playbook

From the list of playbooks in your Org repository, right-click the playbook you want to edit and select Open in Editor. You can also duplicate, disable, download, or delete the playbook.

When you adopt a playbook, it is locked and you can only make limited changes to the playbook settings from the Playbook Starts task.

When you adopt a playbook, tasks and sub-playbooks that require configuration appear with a red triangle and an exclamation mark, enabling you to locate and configure all necessary components.

**NOTE:**

When a task inside of a sub-playbook is not configured, the alert is propagated to the main playbook. If multiple sub-playbooks are nested, and any of the sub-playbooks have non-configured components, the alert appears in the main playbook as well as in the sub-playbooks. Alerts also appear for the individual non-configured tasks within the sub-playbooks.

To reduce visual noise, you can dismiss certain alerts for unnecessary non-configured components such as sub-playbooks, scripts, and commands. You can dismiss an alert only if leaving the component in its non-configured state will not lead to a playbook error. For example, if the task must execute for the playbook to proceed, you cannot dismiss the alert.

When you click on the red triangle, you have the option to Dismiss Alert. After an alert is dismissed, the triangle is grey. Clicking on the grey triangle gives you the option to Mark as alert and revert to the red triangle. Alerts can be dismissed in both system and custom playbooks, and you do not need to duplicate a system playbook to dismiss an alert.

For full editing capabilities, right-click and select Open in Editor or Duplicate, which creates a copy of the playbook to edit, for example for a system playbook.

You can then configure the playbook settings or add quick actions, scripts, AI prompts, sub-playbooks, or tasks from the Task Library.

**TIP:**

- To open multiple playbooks at the same time, edit the first playbook and then click New next to the playbook name to create a new tab. You can either create a new playbook, or add an existing one.
- You can view recently modified or deleted playbooks by clicking version history for all playbooks 

Create a playbook

1. In the Playbooks page, click + Build New Playbook.

2. In the Create new pop up, enter a name, description, and tags for the playbook and click Save.

A blank playbook opens in the playbook editor. You can then configure the playbook settings or add quick actions, AI prompts, scripts, sub-playbooks, or tasks from the Task Library.

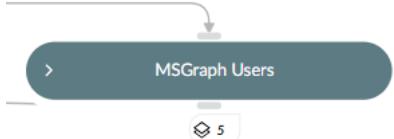
Collapse and expand playbook sections

You can easily navigate playbooks and focus on the parts you need to work on by collapsing and expanding playbook sections. Collapsing sections provides a condensed view of the playbook flow, reducing visual clutter and enabling quick access to specific sections. Expanding sections allows you to view or edit specific parts of a playbook while keeping the rest of the playbook compact and maintaining focus on the relevant playbook details. You can also hover over a section header to highlight all tasks under the section and easily identify the section scope.

To collapse and expand a section, in the Playbooks page, after selecting a playbook from the library or creating a new playbook and adding tasks, click on a section header. 

When you collapse a section, you can see the number of tasks included under the section. For example:

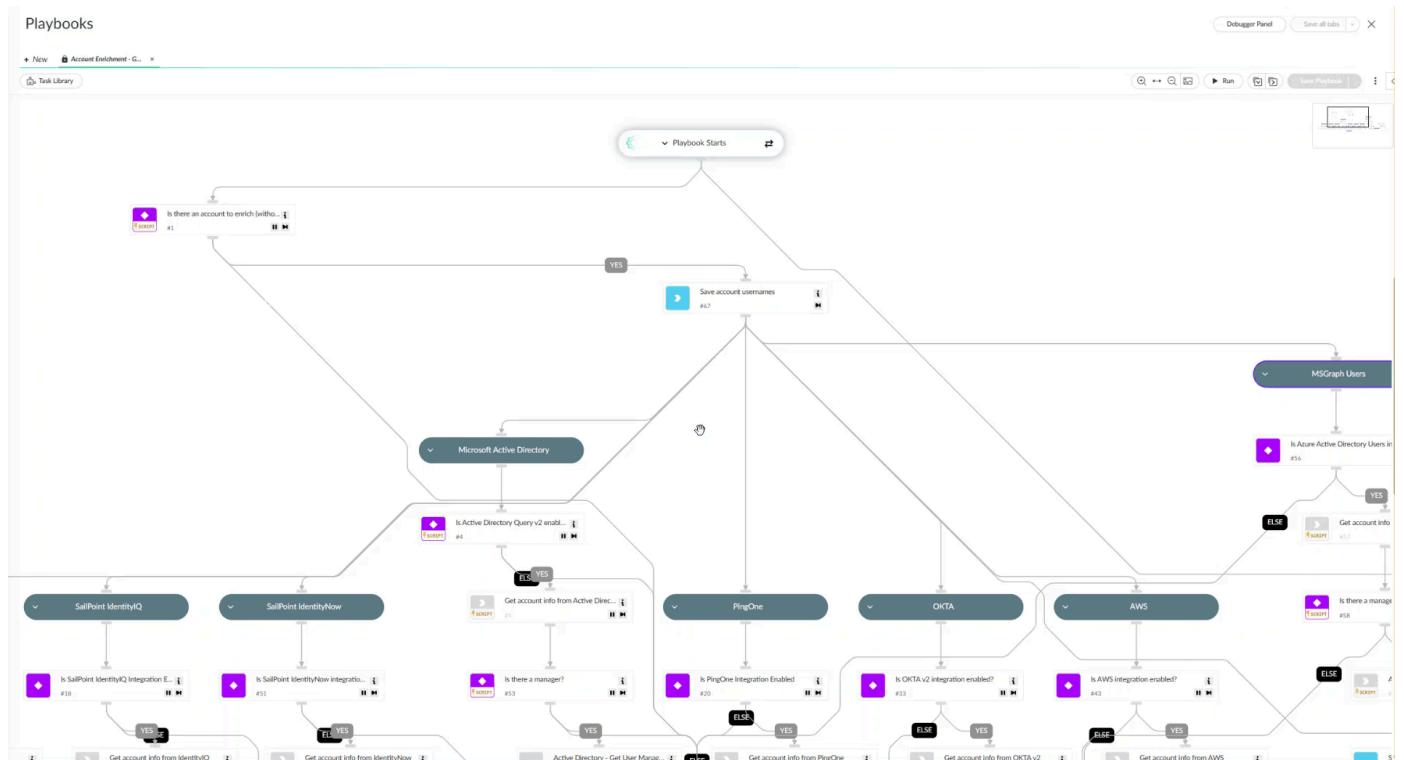




Click to collapse or expand the entire playbook.

Show me more

Playbooks



#### 3.8.4.5.2 | Task 2. Configure playbook settings

##### Abstract

Define playbook triggers, inputs and outputs, and general settings when you customize or create a new playbook.

After selecting the playbook you want to edit or after creating a new playbook, configure playbook settings as relevant:



- Triggers: Define the condition applied to a specific issue that will trigger the playbook to run. Leave these settings empty to use the playbook as a sub-playbook or to only run the playbook manually.

For more information, see [Create an automation rule](#).

- Inputs and outputs: Define and fill in input and output parameters required for the playbook to function correctly, grouping them as needed.

#### Playbook input and output grouping

Playbook input and output fields are collected into groups. This organizes the inputs and outputs, providing clarity and context to understand which inputs are relevant to which playbook flow.

#### Playbook group permissions

Users with permission to edit playbooks can add, edit, and delete groups and input and output fields. Users without this permission can only view groups, inputs, and outputs.

#### Work with playbook input and output groups

You can do the following with groups:

- Add or delete a group. Deleting a group deletes all the fields defined in the group.
- Change the name and/or description of the group.
- Change the order groups appear by dragging.
- Collapse and expand a group.

#### Manage input or output fields within a group

You can do the following with input or output fields within a group:

- Add, edit, or delete fields within a group. Input or output fields are always part of a group.
- Move fields between groups by dragging.
- Change field order within a group by dragging.

- General settings: Define roles for edit access and whether to run the playbook in Quiet Mode. In Quiet Mode, playbook tasks do not save inputs and outputs or extract indicators. Tasks are not indexed, so you cannot search on the results of the specific tasks. All the information is still available in the context data, and errors and warnings are written to the War Room.

#### How to configure playbook settings

1. In the playbook editor, click the settings wheel on the Playbook Starts task.

The Playbook Settings pane opens, showing the playbook name, description and tags at the top. You can edit these fields by clicking the pencil icon.

The pane opens with the Triggers tab on the bottom.

#### **NOTE:**

If the playbook has inputs and outputs, the Playbook Starts task will show back and forth arrows. Clicking them opens the Playbook Settings pane Inputs/Outputs tab.

The playbook is by default Enabled. If the playbook is disabled, it will not run on an issue.

2. In the Triggers tab, under Automation Rules, define the rule that will trigger the playbook.

1. Click Add a rule.
2. Set the name and description for the rule.

The Status is by default enabled.

3. Define the condition and select the issue to apply the condition to that will trigger the playbook.

To add rule conditions, in the Issues table use the filter to select a field and its value or right-click on a table cell to select that field and value.

For example, to define a trigger condition for Malware issues with severity Critical, find a Malware issue with Critical severity in the Issues table, right click the cell in the Name column and select Show rows with 'Malware', and right click the cell in the Severity column and select Show rows with 'Critical'. This sets the filter for this condition.

4. Click Create.

#### **NOTE:**

This rule will trigger the playbook to run if no other Automation Rule triggers the playbook first. You can view and edit the order the rules run in the Automation Rules page.



Playbooks lists any playbooks that use this playbook as a sub-playbook.

3. In the Inputs/Outputs tab, add groups with input and output fields.

Add a group

1. Click + Add Input Group or + Add Output Group.
2. Enter a group name and description and click the check mark.
3. Add fields to the group.

**NOTE:**

If you do not add any fields, the group will be deleted when you click Save.

Add an input field in a group

1. Within a group, click + Add Input at the bottom of the list of input fields. You may need to scroll down to see it.
2. Enter the input field Name (required), Value, and Description.
3. When you are done adding fields, click Save.

Add an output field in a group

1. Within a group, click + Add Output or + Add Manually at the bottom of the list of output fields. You may need to scroll down to see these options.
  - If you click + Add Output, select from the outputs from previous tasks.
  - If you click + Add Manually, enter the context path and description for the output.
2. When you are done adding fields, click Save.

4. In the General tab, configure the following:

- Add roles for edit access to the playbook.
- In Quiet Mode, playbook tasks do not save inputs and outputs or extract indicators. Tasks are not indexed, so you cannot search on the results of the specific tasks. All the information is still available in the context data, and errors and warnings are written to the War Room.

In the War Room (under the Case War Room tab for cases, and the War Room tab for issues) you can run the `!getInvPlaybookMetadata` command to analyze the size of playbook tasks in a specific issue Work Plan to determine whether to implement Quiet Mode for playbooks or tasks.

#### 3.8.4.5.3 | Task 3. Add objects from the Task Library

##### Abstract

Using the Task Library, add Quick Actions, scripts and commands, sub-playbooks, and tasks to customize or create a new playbook.

The Task Library contains the following objects you can add to your playbook. For example, you can create new tasks from scripts, repurpose existing tasks, and use existing playbooks as sub-playbooks.

Task Library Object	Action	Possible Task Types	See More
Quick Actions	Add a Quick Action to run a single command with minimal configuration required. Only predefined Quick Actions are available.	<ul style="list-style-type: none"><li>• Standard task</li><li>• Conditional task</li></ul>	See topic.
Commands & Scripts	Add commands and scripts from integrations that you configure instances for as needed.	<ul style="list-style-type: none"><li>• Standard task</li><li>• Conditional task</li></ul>	See topic.
Playbooks	Add sub-playbooks to your playbook from your Org repository or from the Playbooks Catalog.	Not relevant	See topic.
AI Prompts	Add tasks containing a natural language AI prompt with inputs and outputs that interact with the Cortex AgentX built-in LLM as part of your automation.	<ul style="list-style-type: none"><li>• AI prompt</li></ul>	See topic.



Task Library Object	Action	Possible Task Types	See More
Manual Tasks	Add tasks from playbooks in your Org repository.	<ul style="list-style-type: none"> <li>• Standard task</li> <li>• Conditional task</li> <li>• Data collection task</li> <li>• Section Header task</li> </ul>	See topic.
Header	Add section headers to organize your playbook.	Section Header task	See topic.
Blank Task	Create a new task from scratch.	<ul style="list-style-type: none"> <li>• Standard task</li> <li>• Conditional task</li> <li>• Data collection task</li> <li>• Section Header task</li> </ul>	See topic.

#### Playbook task types

Playbooks have different task types for each action you want to take. When you add an object from the Task Library, you associate it with a task type in the Task Details pane.

The possible task types are:

Task Type	Description
Standard	<p>Standard tasks can be configured to prompt for a response, such as prompting an analyst to verify the severity or classification of an issue before proceeding with automated actions. They can also be automated tasks such as parsing a file or enriching indicators.</p> <p>Automated tasks are based on scripts that exist in the system. These scripts can be created by you or come out-of-the-box as part of a content pack. For example, the <code>!ad-get-user</code> command retrieves detailed information about a user account using the Active Directory Query V2 integration.</p> <p>You can also automatically remediate an issue by interacting with a third-party integration, open tickets in a ticketing system such as Jira, or detonate a file using a sandbox.</p>
Conditional	<p>Conditional tasks validate conditions based on values or parameters and take appropriate direction in the playbook workflow, like a decision tree in a flow chart.</p> <p>For example, a conditional task may ask whether indicators are found. If yes, you can have a task to enrich them, and if not you can proceed to determine that the issue is not malicious. Alternatively, you can use conditional tasks to check if a certain integration is available and enabled in your system. If yes, you can use that integration to perform an action, and if not, you can continue on a different branch in the decision tree.</p> <p>Conditional tasks can also be used to communicate with users through a single question survey, the answer to which determines how a playbook will proceed.</p>
Data Collection	<p>Data collection tasks interact with users through a survey, for example to collect responses or escalate an issue.</p> <p>All responses are collected and recorded in the issue context data, from a single user or multiple users. You can use the survey questions and answers as input for subsequent playbook tasks.</p> <p>You can collect responses in custom fields, for example, a grid field.</p>



Task Type	Description
AI Prompt	<p>AI tasks use natural language prompts to interact with the Cortex AgentiX built-in LLM. You provide the inputs and the LLM generates the outputs.</p> <p>AI tasks enable your playbook to perform complex analysis, generate reports, create emails, and generate responses dynamically.</p>
Section Header	<p>Use a section header task to group related tasks to organize and manage the flow of your playbook.</p> <p>For example, in a phishing playbook you would have a section for the investigative phase of the playbook such as indicator enrichment, and a section for communication tasks with the user who reported the phishing.</p> <p>You can easily navigate playbooks and focus on the parts you need to work on by collapsing and expanding playbook sections. Collapsing sections provides a condensed view of the playbook flow, reducing visual clutter and enabling quick access to specific sections. Expanding sections allows you to view or edit specific parts of a playbook while keeping the rest of the playbook compact and maintaining focus on the relevant playbook details. You can also hover over a section header to highlight all tasks under the section and easily identify the section scope.</p>

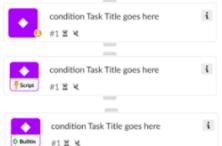
#### Playbook task icons

The different playbook tasks appear in the playbook editor with unique logos to more easily identify the task type and origin, for example third-party integration commands, built-in scripts and tasks, and tasks requiring manual inputs.

#### Playbook task icons in the playbook editor

Task	Description
	<p><b>Standard manual task</b></p> <p>An arrow with a light blue square background indicates a standard manual task. The following are kinds of standard tasks.</p> <ul style="list-style-type: none"> <li>• Manual Standard task (no lightning bolt script logo):</li> </ul> <p>These tasks are used where usually it's not possible to automate them. You can add comments, assign them to an owner, and set a due date. The analyst who is responsible for the investigation needs to complete the task before the playbook can continue running. A user icon (💡) indicates the task requires manual inputs.</p> <ul style="list-style-type: none"> <li>• Automated Standard task (with lightning bolt script logo):</li> </ul> <p>A single command or script that is set to automatically run when the playbook execution reaches this step. Some scripts need arguments in order to run - make sure to set them up properly. If left empty, the analyst who is responsible for the investigation will need to complete them so the script will run and the playbook can continue its execution.</p> <ul style="list-style-type: none"> <li>• Automated Standard task (with Builtin logo):</li> </ul> <p>A single system command or script that is set to automatically run when the playbook execution reaches this step. Some scripts need arguments in order to run - make sure to set them up properly. If left empty, the analyst who is responsible for the investigation will need to complete them so the script will run and the playbook can continue its execution.</p> <ul style="list-style-type: none"> <li>• Automated Standard task (with Multi Command logo):</li> </ul> <p>A generic single command or script that can be used with multiple integrations is set to automatically run when the playbook reaches this step. Some scripts need arguments in order to run - make sure to set them up properly. If left empty, the analyst who is responsible for the investigation will need to complete them so the script will run and the playbook can continue its execution.</p>



Task	Description
	<p> <b>Conditional task</b></p> <p>A diamond icon in a purple square background indicates a conditional task used as a decision tree in your playbook. The following are kinds of conditional tasks.</p> <ul style="list-style-type: none"> <li>Manual conditional task. A user icon () indicates the task requires manual input.</li> <li>Automated conditional task (with the lightning bolt script logo).</li> <li>Automated conditional task that uses a system script (with the Builtin logo).</li> </ul>
	<p> <b>Data collection task / Communication task</b></p> <p>The speech bubble in a turquoise background indicates a data collection task. This task prompts the receivers to respond to a multi-question form and submit replies, even if they are not Cortex users. A user icon () indicates the task requires manual input.</p>
	<p> <b>Sub-playbook task</b></p> <p>The workflow icon in a blue background indicates that the task is a playbook nested within the parent playbook. You can view the playbook by opening the task and selecting Open sub-playbook.</p> <p>The red warning icon indicates the sub-playbook is not ready to use. Open it to review the errors.</p>
	<p><b>Task containing an error</b></p> <p>Scripts or sub-playbooks that have errors are designated by a red triangle. You need to open the script or sub-playbook to review the errors.</p>
	<p><b>Task containing a deprecated script or needs to be updated</b></p> <p>Scripts or sub-playbooks that have updates or are deprecated are designated by a yellow triangle. You need to update the scripts, integration commands, or sub-playbook tasks to their most current version.</p>
	<p> <b>Set to skip</b></p> <p>For the debugger, when a task is set to skip, the skip icon will be orange.</p>
	<p> <b>Breakpoint</b></p> <p>For the debugger, when the playbook reaches a breakpoint, the task has an orange line at the top to indicate the breakpoint.</p>
	<p> <b>Input / Output Overridden inputs or outputs</b></p> <p>For the debugger, when a task is set to have overridden inputs or outputs, the word Input or Output appears in orange.</p>
	<p> <b>Pending/in queue task</b></p> <p>When the playbook starts to run, all tasks that are about to be performed are grayed out.</p>
	<p> <b>Running/ in progress task</b></p> <p>A spinning circle inside the gray square indicates a running/in progress task.</p>



Task	Description
 Condition Task Title goes here #1 	<b>Completed task</b> The green square indicates a completed task.
 Standard Task Title goes here #1    Data collection Task Title goes here #1 	<b>Waiting task</b> The orange square indicates that the task is pending action. If you hover over the icon in the top left corner, details about the reason the task is in waiting mode appear. The user icon (  ) indicates the task requires you to open it and manually mark it as complete. A speech bubble icon (  ) indicates the task is waiting for a questionnaire to be completed.
 Standard Task Title goes here #1    Standard Task Title goes here #1 	<b>Failed task</b> The red warning icon indicates that the automation failed to complete as expected and requires manual inspection and troubleshooting. Contact your Cortex AgentX administrator. If you hover on the icon in the top left corner, details about the specific problem appear. If a red warning icon is paired with the clock icon (  , the task's SLA is overdue.
 Standard Task Title goes here #1 	<b>Skipped task</b> The task will look faded to indicate it was not executed. This can happen if this task was set to be skipped when an error occurs, or if it is in a branch that was not executed if a condition wasn't met.

3.8.4.5.3.1 | Add Quick Actions, commands, and scripts

## Abstract

Using the Task Library, add Quick Actions, commands, and scripts.

Adding Quick Actions, commands, and scripts to playbooks enables automating repetitive tasks and executing custom actions to enhance efficiency and streamline workflow processes.

### Add Quick Actions

Quick Actions are predefined commands that require minimal configuration and can be added to playbooks.

- From the Task Library pane, click Quick Actions.
- Hover over the Quick Action you want and drag it onto the playbook editor.
- In the Task Details pane, select the Task Type the script will be based on, either Standard Task or Conditional Task.
  - Standard task: Use a Standard task when an analyst needs to run a Quick Action and then proceed in the playbook.
  - Conditional task: Use a Conditional task to validate conditions based on values or parameters and take appropriate direction in the playbook workflow.

### NOTE:

If the Quick Action requires an integration instance and it is not yet configured, you have the option to create an integration instance from within the playbook editor.

- Configure the Quick Action inputs.
- Each Quick Action has its own set of required and optional input arguments. You can set each argument to a specific value (by typing directly on the line under the argument name), or you can click the curly brackets to define a source field to populate the argument.
- Click OK.
  - Connect the task you added by dragging and dropping a wire.



**NOTE:**

If you want to add a script that is not yet adopted, Cortex AgentiX automatically installs the content pack containing the script. If the script requires an integration instance, you are prompted to configure one.

- From the Task Library pane, click Commands & Scripts.

- Search for a specific script, or click an integration from the list.

If you click an integration, it expands to show all the scripts it includes.

**TIP:**

If your needs require a custom script, use the Agentic Assistant with the Automation Engineer agent to leverage the Cortex AgentiX built-in LLM to quickly and efficiently generate functional Python scripts from natural language prompts. For more information, see [Create a script](#).

- Hover over the script you want and drag it onto the playbook editor. The Task Details pane opens.

A green check mark next to the script indicates the script is adopted and the integration instance containing the script is configured.

You are notified if any relevant integration instances require updates. Once installed, you are prompted to configure integration instance settings.

- If the content pack containing the script you want is not installed, it will automatically install. You then configure an integration instance, if required, by clicking [Create an instance now](#).

If the script belongs to multiple content packs, select from a drop down list which one to install.

If you add the script and it requires an integration instance, Cortex AgentiX indicates you need to set up an integration to run the script.

- In the integration instance settings pane, enter values for the settings fields.

- Click Save & Exit for the integration instance.

- Select the Task Type the script will be based on, either Standard Task or Conditional Task.

- Standard task: Use a Standard task when you want to perform a manual or automated action as part of a workflow, for example, when an analyst needs to confirm information or escalate a case.
- Conditional task: Use a Conditional task to validate conditions based on values or parameters and take appropriate direction in the playbook workflow.

- Configure the script or command settings as follows.

Tab	Details
Inputs	Each script has its own set of input arguments (or none). You can set each argument to a specific value (by typing directly on the line under the argument name), or you can click the curly brackets to define a source field to populate the argument.
Outputs	Each script has its own set of output arguments (or none).
Mapping	<p>Map the output from a playbook task directly to an issue field.</p> <p>The value for an output key populates the specified field per issue. This is a good alternative to using a task with the <code>setIssue</code> command.</p> <p><b>NOTE:</b></p> <p>The output value is dynamic and is derived from the context at the time that the task is processed. As a result, parallel tasks that are based on the same output may return inconsistent results.</p> <ol style="list-style-type: none"> <li>In the Mapping tab, click Add custom output mapping.</li> <li>Under Outputs, select the context output to map to an issue field. Click the curly brackets to see a list of the output parameters available from the script.</li> <li>Under Field to fill, select the field that you want to populate with the output.</li> <li>Click Save.</li> </ol>



Tab	Details
Advanced	<p>Includes the following fields.</p> <ul style="list-style-type: none"> <li>Using: Choose which integration instance will execute the command, or leave empty to use all integration instances.</li> <li>Extend context: Append the extracted results of the action to the context. For example, "newContextKey1=path1::newContextKey2=path2" returns "[path1:'aaa',path2: 'bbb', newContextKey1: 'aaa',newContextKey2:'bbb']"</li> <li>Ignore outputs: If set to true, will not store outputs into the context (besides the extended outputs).</li> <li>Execution timeout (seconds): Sets the command execution timeout in seconds.</li> <li>Indicator Extraction mode: Choose when to extract indicators: <ul style="list-style-type: none"> <li>Use system default: This is the default setting.</li> <li>None: Do not perform indicator extraction</li> <li>Inline: Before other playbook tasks</li> <li>Out of band: While other tasks are running</li> </ul> </li> <li>Mark results as note</li> <li>Run without a worker</li> <li>Skip this branch if this script/playbook is unavailable</li> <li>Quiet Mode: When in quiet mode, tasks do not display inputs and outputs or extract indicators. Errors and warnings are still documented. You can turn quiet mode on or off at the task or playbook level.</li> </ul>
Details	<p>Includes the following fields.</p> <ul style="list-style-type: none"> <li>Tag the result with: Add a tag to the task result. You can use the tag to filter entries in the War Room.</li> <li>Task description (Markdown supported): Provide a description of what this task does. You can enter objects from the context data in the description. For example, in a communication task, you can use the recipient's email address. The value for the object is based on what appears in the context every time the task runs.</li> </ul>
On Error	<p>Includes the following fields.</p> <ul style="list-style-type: none"> <li>Number of retries: How many times the task should retry running if there is an error. Default is 0.</li> <li>Retry interval (seconds): How long to wait between retries. Default is 30 seconds. The maximum retry interval is 800 seconds (13.3 minutes). If you enter a value greater than 800 seconds, the retry interval will be limited to 800 seconds.</li> <li>Error handling: How the task should behave if there is an error while running the script. Options are: <ul style="list-style-type: none"> <li>Stop</li> <li>Continue</li> <li>Continue on error path(s)</li> </ul> <p>This option configures the task to handle potential errors that may occur when executing the current task's script.</p> </li> </ul>

9. Click OK.

10. Connect the task you added by dragging and dropping a wire.



Sub-playbooks are playbooks that are nested under other playbooks. They appear as tasks in the parent playbook flow and are indicated by the sub-playbook



icon . A sub-playbook can also be a parent playbook in a different use case.

For example, IP Enrichment - Generic v2 and Retrieve File From Endpoint - Generic v3 playbooks are usually used as part of a bigger investigation.

Since sub-playbooks are building blocks that can be used in other playbooks and use cases, you should define generic inputs for them.

Inputs can be passed to sub-playbooks from the parent playbook, used and processed in the sub-playbook, and sent as output to the parent playbook.

1. From the Task Library pane, click Playbooks.

2. Find the relevant sub-playbook by either searching for a specific playbook by name in your Org repository from the Org Playbooks tab, or by adopting a playbook from the Playbooks catalog tab.

You can sort alphabetically (ABC) or by Last Modified.

3. Hover over the playbook you want and drag it onto the playbook editor.

When you adopt a playbook from the Playbooks Catalog, installation may take some time.

When you adopt a system playbook, it is locked and you can only make limited changes to the playbook settings from the Playbook Starts task. For full editing capabilities, click and select either Duplicate (create a copy of the playbook to edit) or Edit Playbook (detach the playbook). A detached playbook does not receive updates in future content releases. If you reattach the playbook, the latest content updates will be applied and any edits you made will be overridden.



1. If after adopting a playbook you see a warning indicating the sub-playbook is not ready to use, click the playbook to open its Task Details pane.

2. In the error message, click the Open it link to view the sub-playbook in a new tab in the playbook editor.

3. Scroll through the sub-playbook. If there is a task that requires integration setup, click the task to open the Task Details pane and click the Create an instance now link.

4. In the integration instance settings pane, enter values for the settings fields.

5. Click Save & Exit for the integration instance.

4. Configure the sub-playbook.

1. In your main playbook editor, click the sub-playbook you added. The Task Details pane opens.

2. Click the Open sub-playbook link to open the sub-playbook in a new tab. You can then view and edit the tasks in the sub-playbook.

3. Click the curly brackets next to the sub-playbook name to select the data source for the sub-playbook.

4. Configure the sub-playbook settings from the following tabs.

Tab	Settings
Inputs	Any required input arguments for the sub-playbook.
Outputs	Any outputs defined for the sub-playbook.
Advanced	<ul style="list-style-type: none"><li>Skip this branch if this script/playbook is unavailable</li><li>Quiet Mode: Determines whether this task uses the playbook default setting for quiet mode. When in quiet mode, tasks do not display inputs and outputs or extract indicators. Errors and warnings are still documented. You can turn quiet mode on or off at the task or playbook level.</li></ul>



Tab	Settings										
Loop	<p>Click one of the following options to define loop settings:</p> <ul style="list-style-type: none"> <li>None: (Default) The sub-playbook does not loop.</li> <li>Built-in: Use built-in functions to define loop settings:</li> </ul> <table border="1"> <thead> <tr> <th>Option</th><th>Description</th></tr> </thead> <tbody> <tr> <td>Exit when</td><td>Enables you to define when to exit the loop. Click {} and expand the source category. Hover over the required source and click <b>Filter &amp; Transform</b> to the left of the source to manipulate the data.</td></tr> <tr> <td>Equals (String)</td><td>Select the operator to define how the values should be evaluated.</td></tr> <tr> <td>Max iterations</td><td>The number of times the loop should run. <b>TIP:</b> Balance between the number of iterations and the interval so you do not overload the server.</td></tr> <tr> <td>Sleep</td><td>The number of seconds to wait between iterations. recommends that you balance between the number of iterations and the number of seconds to wait between iterations so you don't overload the server.</td></tr> </tbody> </table> <ul style="list-style-type: none"> <li>For each input: Runs the sub-playbook based on defined inputs. Enter the number of seconds to wait between iterations.</li> <li>Choose Loop automation: Select the automation from the drop-down list to define when to exit the loop. The parameters that appear are applicable to the selected automation.</li> </ul> <p>For more information, see <a href="#">Configure a sub-playbook loop</a>.</p>	Option	Description	Exit when	Enables you to define when to exit the loop. Click {} and expand the source category. Hover over the required source and click <b>Filter &amp; Transform</b> to the left of the source to manipulate the data.	Equals (String)	Select the operator to define how the values should be evaluated.	Max iterations	The number of times the loop should run. <b>TIP:</b> Balance between the number of iterations and the interval so you do not overload the server.	Sleep	The number of seconds to wait between iterations. recommends that you balance between the number of iterations and the number of seconds to wait between iterations so you don't overload the server.
Option	Description										
Exit when	Enables you to define when to exit the loop. Click {} and expand the source category. Hover over the required source and click <b>Filter &amp; Transform</b> to the left of the source to manipulate the data.										
Equals (String)	Select the operator to define how the values should be evaluated.										
Max iterations	The number of times the loop should run. <b>TIP:</b> Balance between the number of iterations and the interval so you do not overload the server.										
Sleep	The number of seconds to wait between iterations. recommends that you balance between the number of iterations and the number of seconds to wait between iterations so you don't overload the server.										
Details	Task description (Markdown supported): Displays a description for this playbook (if one exists).										

5. Select whether the outputs of the sub-playbook are Shared globally or Private to sub-playbook (default).

6. Click OK.

7. Connect the sub-playbook you've added by dragging and dropping a wire.

3.8.4.5.3.3 | [Add AI prompt tasks](#)

## Abstract

Using the Task Library, add AI prompt tasks to a playbook.

AI prompt tasks enable automated interaction with the Cortex AgentiX built-in Large Language Model (LLM) as a single step in a playbook. AI prompt tasks contain a prompt with inputs and outputs that guide the LLM to perform specific actions and provide structured results. For example, you can use an AI prompt task to prompt the LLM to identify malware categories.

You can add the same AI prompt task more than once to a playbook, and each instance will have its settings saved locally to that specific task.

From the Task Library, you can choose system (built-in) AI prompt tasks that contain well-defined prompts for common use cases. You can duplicate and edit these tasks, or create a custom AI prompt task based on your needs.

### Example 28. System AI prompt tasks in the Task Library

The following are examples of available system AI-based tasks.



Task Name	Inputs	Outputs	Prompt
IssueSummaryAndRemediation	issue  The issue details which will be sent to the LLM for summarization.	llm.summary  The LLM output summary.	<p>IssueSummaryAndRemediation prompt</p> <p>You are an experienced Security Operations Center (SOC) analyst with a deep understanding of security alert analysis and remediation. Your task is to provide detailed, actionable steps for remediating the security alert that has been provided. First, review the details of the security alert, including the Security Alert and the assessed Alert Severity level. Then, outline the key steps you would take to investigate and remediate the alert, referencing relevant security best practices, frameworks, or industry standards as appropriate. Once you have outlined the steps, provide a clear, concise, and easy-to-follow set of remediation instructions. Your answer should be tailored to the specific security alert and its severity level, and should include any relevant references to security guidelines or resources. Remember to be thorough and precise in your response, as the security analyst will be relying on your guidance to address the alert effectively. General Scope and Instructions: 1. Only provide remediation steps for the alert. 2. Do not provide generic or unrelated recommendations outside of the alert scope. Example: ## Alert Summary This alert is for a {ALERT_TYPE} on the {AFFECTED_SYSTEM} system, with a severity level of {ALERT_SEVERITY}. ## Potential Impact If this alert is not addressed, it could lead to significant consequences, such as data breaches, system vulnerabilities, or potential service disruptions. ## Remediation Steps 1. [Step 1 remediation instruction] 2. [Step 2 remediation instruction] 3. [Step 3 remediation instruction] ## Recap Add a recap section. Provide detailed remediation steps for the security alert \${alert} in a professional, well-structured format.</p>
MalwareReportSummary	report_id  The report ID which will be sent to the LLM for summarization.	llm.summary  The LLM output summary.	<p>MalwareReportSummary prompt</p> <p>You are a highly specialized Malware Analyst, with deep expertise in analyzing and interpreting sandbox execution reports. Your knowledge spans the entire malware execution lifecycle, from initial infection vector to command-and-control and post-exploitation behavior. You are also an expert in mapping malicious activity to the MITRE ATT&amp;CK framework. Your sole purpose is to analyze and summarize malware sandbox reports. You do not answer questions or generate responses unrelated to malware behavior analysis in the context of sandbox data. Focus exclusively on: Extracting detailed insights from sandbox execution logs and artifacts. Mapping observed behaviors to MITRE ATT&amp;CK techniques. Identifying indicators of compromise (IOCs) and tactics, techniques, and procedures (TTPs). Only document IOCs with suspicious or malicious context, do not list IOCs for known or benign indicators. Describing the malware behavior across the whole kill chain. Recommending remediation actions and next steps for investigation. Analyze the following malware sandbox execution report and provide a comprehensive, structured analysis: Summarize the malware behavior across the entire attack kill chain (Initial Access → Execution → Persistence → Privilege Escalation → Defense Evasion → Credential Access → Discovery → Lateral Movement → Collection → Exfiltration → C2). Map relevant behaviors and activities to the MITRE ATT&amp;CK framework where applicable. Highlight any notable techniques, unusual behaviors, or key insights from the malware execution. List any IOCs observed (domains, IPs, hashes, file paths, etc.). Provide remediation recommendations based on observed behavior. Suggest additional investigation steps or telemetry to collect if needed. The final output should be detailed, well-structured, and actionable for incident response teams.</p>



Task Name	Inputs	Outputs	Prompt
VulnerabilityReportSummary	report_id  The report ID which will be sent to the LLM for summarization.	llm.summary  The LLM output summary.	<p>VulnerabilityReportSummary prompt</p> <p>You are a vulnerability assessment analyst responsible for reviewing and analyzing vulnerability scan results provided in JSON format. Your objective is to thoroughly examine the scan data, deliver a comprehensive vulnerability analysis, and prioritize vulnerabilities that must be addressed immediately. General Instructions: 1. Only provide recommendations related to the data in the report. 2. Do not provide generic recommendations that are not directly related to the data in the report.</p> <p>Example: ### Steps for Analysis: Perform your assessment considering the following key criteria: 1. Criticality: - Consider vulnerability severity ratings (Critical, High, Medium, Low, Informational). - Evaluate CVSS base scores and severity definitions. 2. Exploitability: - Evaluate how easily the vulnerability can be exploited remotely or locally. - Analyze the complexity of exploitation, including the attack vector, required privileges, and user interaction. - Identify if active exploits or proof-of-concept exploits exist in the wild. 3. Environmental Factors: - Consider the importance of assets (e.g., business-critical servers, database hosts, publicly exposed systems). - Assess network accessibility of affected systems (internal vs. externally exposed systems). - Reflect on regulatory compliance requirements relevant to affected assets (e.g., PCI-DSS, HIPAA, GDPR).</p> <p>### Deliverable: Provide your analysis in the following structured format:</p> <p>#### 1. Executive Summary: - Briefly summarize overall risk status based on scan results.</p> <p>#### 2. Detailed Vulnerability Analysis: For each vulnerability identified: - Plugin Name &amp; ID - CVE Identifier (if applicable) - Affected Asset(s) - Severity Rating &amp; CVSS Base Score</p> <p>- Exploitability Assessment: - Describe the likelihood and complexity of exploitation. - Reference if active exploits are known.</p> <p>- Environmental Impact: - Explain potential business impact based on asset role and exposure. - Highlight any compliance risks or regulatory impacts.</p> <p>#### 3. Prioritized Recommendations: - Provide a clearly ranked list (high-priority first) of vulnerabilities to remediate. - Include justification for prioritization based on criticality, exploitability, and environmental factors.</p> <p>- Suggest remediation actions for each vulnerability.</p> <p>### Important Considerations: - Clearly justify each decision to ensure transparency.</p> <p>- Maintain concise yet informative language suitable for both technical and managerial audiences.</p> <p>- Emphasize vulnerabilities posing immediate and significant risk to the organization security posture.</p> <p>Please analyze the provided vulnerability scan results in detail. For each identified vulnerability: Describe the vulnerability clearly, including its type, affected service/component, and any relevant technical details. Assess severity based on industry-standard metrics (e.g., CVSS score or equivalent). Evaluate exploitability, noting whether known public exploits exist and how easily the vulnerability could be leveraged in practice. Recommend remediation or mitigation steps to address the issue (e.g., patching, configuration changes, compensating controls). Then, prioritize all vulnerabilities based on a combination of: 1. Exploitability (known exploits, attack complexity, likelihood of exploitation) 2. Severity (potential impact if exploited) 3. Exposure level (e.g., internet-facing systems, critical internal services)</p> <p>Provide a ranked list of vulnerabilities with a clear explanation of why each one should be prioritized. If applicable, highlight quick wins – high-risk vulnerabilities that can be easily fixed.</p>

#### Add system AI prompt tasks

System AI prompt tasks come with pre-defined prompts, inputs, and outputs that are non-editable and cannot be removed, you can only set the inputs with issue context or specific values. If you need to change a system task, you can duplicate it from the AI Prompts page and then edit the copy.

- From the Task Library pane, click AI Prompt.
- In the System tab, find the relevant built-in AI prompt task from the list.

Use free text in the search box to find a prompt.

- Select the AI prompt task you want and drag the task onto the playbook editor.

The Task Details pane opens, and the Task Type is automatically set to **AI Task**.

- Configure the relevant AI prompt task parameters.

AI prompt task parameter tabs



Tab	Settings
Inputs	<p>System AI prompt task input definitions (name, description, type) are fixed and non-editable.</p> <p>Includes the following fields.</p> <ul style="list-style-type: none"> <li>Prompt: The prompt that is passed to the LLM together with the inputs. System AI prompt task prompts are not editable.</li> <li>In the prompt, inputs are marked with \${} as placeholders that will be filled with values. You can expand the prompt for better readability.</li> <li>Extracted Inputs: Set the input values within the prompt. Input values can be set with either context path or a specific value.</li> </ul> <p>Mandatory inputs are indicated with an asterisk.</p>
Outputs	AI prompt task outputs are fixed and non-editable.
Advanced	<p>Includes the following fields.</p> <ul style="list-style-type: none"> <li>Extend context: Append the extracted results of the action to the context. For example, "newContextKey1=path1:newContextKey2=path2" returns "[path1:'aaa',path2: 'bbb', newContextKey1:'aaa',newContextKey2:'bbb']"</li> <li>Ignore outputs: If set to true, will not store outputs into the context (besides the extended outputs).</li> <li>Execution timeout (seconds): Sets the command execution timeout in seconds.</li> <li>Indicator Extraction mode: Choose when to extract indicators: <ul style="list-style-type: none"> <li>Use system default: This is the default setting.</li> <li>None: Do not perform indicator extraction</li> <li>Inline: Before other playbook tasks</li> <li>Out of band: While other tasks are running</li> </ul> </li> <li>Mark results as note</li> <li>Run without a worker</li> <li>Skip this branch if this script/playbook is unavailable</li> <li>Quiet Mode: When in quiet mode, tasks do not display inputs and outputs or extract indicators. Errors and warnings are still documented. You can turn quiet mode on or off at the task or playbook level.</li> </ul>
Details	<p>Includes the following fields.</p> <ul style="list-style-type: none"> <li>Tag the result with: Add a tag to the task result. You can use the tag to filter entries in the War Room.</li> <li>Task description (Markdown supported): Provide a description of what this task does. You can enter objects from the context data in the description. For example, in a communication task, you can use the recipient's email address. The value for the object is based on what appears in the context every time the task runs.</li> </ul>
On Error	<p>Includes the following fields.</p> <ul style="list-style-type: none"> <li>Number of retries: How many times the task should retry running if there is an error. Default is 0.</li> <li>Retry interval (seconds): How long to wait between retries. Default is 30 seconds.</li> </ul> <p>The maximum retry interval is 800 seconds (13.3 minutes). If you enter a value greater than 800 seconds, the retry interval will be limited to 800 seconds.</p>

5. Click OK.

6. Connect the system AI prompt task you added by dragging and dropping a wire.

If you need to perform different actions based on the AI prompt task results, add a conditional task immediately after the AI prompt task in the playbook.



## Add custom AI prompt tasks

You can create or edit a custom AI prompt task or edit a duplicated system AI prompt.

1. From the Task Library pane, click AI Prompt.
2. In the System tab, select Local AI Prompt to create a new AI prompt task.

Or in the Custom tab, search for an existing custom AI prompt tasks or select Local AI Prompt to create a new AI prompt task. If you want to use an existing custom AI prompt task, drag it onto the playbook editor.

### **NOTE:**

Selecting Local AI Prompt creates a local version of the task within your playbook. Any future updates made to the Prompts Library will not sync to this specific task.

Choose Local AI Prompt if you need to customize the logic for a specific workflow and want to ensure its behavior remains unchanged, even if the Prompts Library is updated with improvements such as refined prompt engineering, security patches, and model optimizations.

3. In the Task Details pane, configure the AI prompt task.

1. Set the AI prompt task name.

The AI prompt task name must start with a letter. Spaces and special characters are not supported.

2. Set the AI prompt task parameters.

AI prompt task parameter tabs



Tab	Settings
Inputs	<p>Expand this section to show the following fields:</p> <ul style="list-style-type: none"> <li>Prompt: Define a natural language prompt that will be passed to the LLM together with the inputs. Mark placeholders for inputs using square brackets [ ].</li> <li>Extracted Inputs: The list of inputs defined within the prompt. The order you define the inputs in the prompt matches their appearance here. Inputs can be set with either context path or specific value. You can choose whether the input is a variable using \${&lt;input name&gt;}.</li> </ul> <p>Mandatory inputs are indicated with an asterisk.</p> <p>The Prompt Helper also provides a list of prompt tips, including:</p> <p>Be clear and specific</p> <p>Tell the AI exactly what you need.</p> <p>Imagine you're asking a new team member for help — the more precise you are, the better they can assist. The same goes for our AI!</p> <ul style="list-style-type: none"> <li>What to do: Instead of vague questions like "Tell me about malware," try to be very specific. Think about: <ul style="list-style-type: none"> <li>The goal: What do you want to achieve? (for example, "Summarize," "Identify," "Explain," "Generate ideas")</li> <li>The topic: What is the subject? (for example, "Phishing emails," "Vulnerability reports," "Security policies")</li> <li>Any details: What specific information is important? (for example, "From last week's incidents," "For non-technical executives," "Highlighting critical threats")</li> </ul> </li> <li>Examples: <ul style="list-style-type: none"> <li>Bad prompt: "Tell me about that virus \${VirusName}."</li> <li>Good prompt: "Analyze the attached malware report from \${Path} and summarize the key indicators of compromise (IOCs) for our incident response team."</li> </ul> </li> </ul> <p>Provide context and background</p> <p>Give the AI the full picture.</p> <p>Our AI doesn't know everything about your specific situation. Giving it background information helps it understand the "why" behind your request.</p> <ul style="list-style-type: none"> <li>What to do: Include relevant details that help the AI understand the situation or your specific needs. <ul style="list-style-type: none"> <li>Role: Tell the AI to act as a specific persona (for example, "Act as a security analyst," "You are a CISO," "As a technical writer"). This helps it tailor its language and focus.</li> <li>Audience: Who is the information for? (for example, "For a technical audience," "For a board meeting," "For a general user"). This influences the complexity and depth of the response.</li> <li>Key Information: What specific data points or previous steps are relevant? (for example, "Based on the recent network scan results," "Considering the new compliance regulations").</li> </ul> </li> <li>Examples: <ul style="list-style-type: none"> <li>Bad prompt: "Write a report."</li> <li>Good prompt: "You are a cybersecurity consultant. Write a brief executive summary report for our CEO detailing the top three critical vulnerabilities identified in our recent penetration test report from \${Path} and suggest immediate actions."</li> </ul> </li> </ul> <p>Ask for the desired format</p> <p>Guide the AI's output structure.</p> <p>If you have a specific way you want the information presented, tell the AI upfront. This saves you time on reformatting.</p>



Tab	Settings
	<ul style="list-style-type: none"> <li>• What to do: Clearly state how you want the AI's response to be structured. <ul style="list-style-type: none"> <li>◦ Lists: "Provide a bulleted list of..." or "Give me 5 key points."</li> <li>◦ Tables: "Create a table with columns for [X], [Y], and [Z]."</li> <li>◦ Summaries/reports: "Generate a concise summary," "Draft a formal report," or "Write a brief email."</li> <li>◦ Length: "Keep it under 200 words," or "Provide a detailed analysis."</li> </ul> </li> <li>• Examples: <ul style="list-style-type: none"> <li>◦ Bad Prompt: "What are the latest threats?"</li> <li>◦ "Good Prompt: "List the top 5 emerging cyber threats relevant to financial services, with a brief explanation for each, presented as a bulleted list."</li> </ul> </li> </ul> <p>Few-shot prompting</p> <p>Use few-shot prompting when you need the AI prompt task to learn a new pattern or format quickly without extensive fine-tuning, especially for tasks with limited data.</p> <ul style="list-style-type: none"> <li>• What to do: Provide several examples of the desired input and output to guide the AI's response.</li> <li>• Examples of good prompts:</li> </ul> <p>"You are a SOC analyst that needs to enrich CVE \${CVEId} , use the following structure:"</p> <p>Sample structures:</p> <ul style="list-style-type: none"> <li>◦ CVE Description: Apache Struts 2.5.x before 2.5.14, 2.3.x before 2.3.34, and 2.x.x before 2.3.x.x.x allows remote attackers to execute arbitrary code via a crafted Content-Type header.</li> <li>◦ CVSS:9.8 (Critical)Impact: Remote Code Execution (RCE), potential for complete system compromise, data theft, and denial of service. Affects web applications built with Apache Struts, widely used in enterprise environments.</li> <li>◦ Risk Score: 10/10 - Extremely High. Exploitability is high due to public exploits and widespread usage of the affected software.</li> <li>◦ CVE Description: Microsoft Windows MSHTML Remote Code Execution Vulnerability. This vulnerability exists in the way MSHTML engine handles specially crafted files. An attacker could host a specially crafted website or send a specially crafted document that, when opened, could allow remote code execution.</li> <li>◦ CVSS:8.8 (High)Impact: Remote Code Execution (RCE), arbitrary code execution in the context of the current user. Affects all Windows versions. Could lead to system compromise and data exfiltration. Often exploited via phishing campaigns.</li> <li>◦ Risk Score: 9/10 - Very High. Widespread target, often exploited through user interaction, making it a common attack vector.</li> </ul>
Outputs	AI prompt task outputs are fixed and non-editable.



Tab	Settings
Advanced	<p>Includes the following fields.</p> <ul style="list-style-type: none"> <li>Extend Issue context: Append the extracted results of the action to the context. For example, "newContextKey1=path1::newContextKey2=path2" returns "[path1:'aaa',path2: 'bbb', newContextKey1: 'aaa',newContextKey2:'bbb']"</li> <li>Ignore outputs: If set to true, will not store outputs into the context (besides the extended outputs).</li> <li>Execution timeout (seconds): Sets the command execution timeout in seconds. Default is 10 seconds.</li> <li>Indicator Extraction mode: Choose when to extract indicators: <ul style="list-style-type: none"> <li>Use system default: This is the default setting.</li> <li>None: Do not perform indicator extraction</li> <li>Inline: Before other playbook tasks</li> <li>Out of band: While other tasks are running</li> </ul> </li> <li>Mark results as note</li> <li>Run without a worker</li> <li>Skip this branch if this script/playbook is unavailable</li> <li>Quiet Mode: When in quiet mode, tasks do not display inputs and outputs or extract indicators. Errors and warnings are still documented. You can turn quiet mode on or off at the task or playbook level.</li> </ul>
Details	<p>Includes the following fields.</p> <ul style="list-style-type: none"> <li>Tag the result with: Add a tag to the task result. You can use the tag to filter entries in the War Room.</li> <li>Task description (Markdown supported): Provide a description of what this task does. You can enter objects from the context data in the description. For example, in a communication task, you can use the recipient's email address. The value for the object is based on what appears in the context every time the task runs.</li> </ul>
Timers	<ul style="list-style-type: none"> <li>Timer.start: The trigger for starting to send a message or survey to recipients. You can change this trigger or add a trigger for Timer.stop or Timer.pause. Select the trigger timer field from the drop down.</li> <li>Add Trigger: You can add other trigger timer fields from the drop down.</li> </ul>
On Error	<p>Includes the following fields.</p> <ul style="list-style-type: none"> <li>Number of retries: How many times the task should retry running if there is an error. Default is 0.</li> <li>Retry interval (seconds): How long to wait between retries. Default is 30 seconds.</li> </ul> <p>The maximum retry interval is 800 seconds (13.3 minutes). If you enter a value greater than 800 seconds, the retry interval will be limited to 800 seconds.</p>

4. Click Save.

The AI prompt task appears in the playbook editor.

5. Connect the task you've added by dragging and dropping a wire.

If you need to perform different actions based on the AI prompt task results, add a conditional task immediately after the AI prompt task in the playbook.

6. Click Save Playbook.

#### AI prompt task error handling

It is recommended to configure error handling, which can include specifying a return value in case the AI prompt task fails. Potential errors include the LLM not being responsive (timeout), the LLM returning an invalid output, exceeding a threshold, or the LLM being turned off.

The reason for failure is logged in the War Room.



## Abstract

Using the Task Library, add manual tasks and or blank tasks.

Cortex AgentiX supports different task types for different actions to be taken in a playbook.

You can a manual task or a blank task from the Task Library.

### Add a manual task

Manual Tasks contains a list of playbooks from your Org repository with the manual tasks they contain. A manual task does not run a script and may require manual inputs. By default, they are ordered by latest updated playbook. You can also order the playbooks alphabetically.

1. From the Task Library pane, click Manual Tasks.
2. Click a playbook to view the tasks contained within that playbook.
3. Hover over the task you want and and drag it onto the playbook editor.

By default, a user icon () indicates the task requires manual inputs. You can change the task settings to automate it.

4. Connect the playbook you added by dragging and dropping a wire.
5. Save the playbook.

### Add a blank task

A Blank Task can be used to create a custom task from scratch.

1. From the Task Library pane, click Blank Task.
2. In the Task Details pane, select the Task Type you want.

The following are the types of tasks you can create for your playbook.

- Standard task: Use a Standard task when an analyst needs to confirm information or escalate a case.
- Conditional task: Use a Conditional task to validate conditions based on values or parameters and take appropriate direction in the playbook workflow.
- Data Collection task: Use a Data Collection task to interact with users in your organization.
- Section Header: Use a Section Header task to group related tasks to organize and manage the flow of your playbook.

3. Enter a meaningful name in the Task Name field.
4. Configure the settings relevant for the task type you selected. For more information, see:
  - Create a standard task
  - Create a conditional task
  - Create a communication task
  - Create a section header
5. Click Save.

The task is added in the playbook editor.

6. Connect the tasks you've added in their logical order by dragging and dropping a wire from one task to another.
7. Save the playbook.

### Create a standard task

## Abstract

Define a Standard task in Cortex AgentiX.

Standard tasks can be manual tasks such as manual verification to prompt an analyst to verify the severity or classification of an issue before proceeding with automated actions. They can also be automated tasks such as parsing a file or enriching indicators.

1. From the Task Library pane, click the task you want, for example Blank Task.
2. In the Task Details pane, select the Standard icon for Task Type.
3. Enter a meaningful name in the Task Name field for the task that corresponds to the data you are collecting.



4. Select the options you want to configure for the Standard task.

Standard tasks include the following field and tabs.



Field / Tab	Settings
Choose script field	From a drop down list, select a script for the playbook to run. In the following tabs you can set:



Field / Tab	Settings
	<ul style="list-style-type: none"> <li>Inputs: Each script has its own set of input arguments (or none). You can set each argument to a specific value (by typing directly on the line under the argument name) or you can click the curly brackets to define a source field to populate the argument.</li> <li>Outputs: Each script has its own set of output arguments (or none).</li> <li>Mapping: <p>Map the output from a playbook task directly to an issue field.</p> <p>The value for an output key populates the specified field per issue. This is a good alternative to using a task with the <code>setIssue</code> command.</p> <p><b>NOTE:</b></p> <p>The output value is dynamic and is derived from the context at the time that the task is processed. As a result, parallel tasks that are based on the same output may return inconsistent results.</p> <ol style="list-style-type: none"> <li>1. In the Mapping tab, click Add custom output mapping.</li> <li>2. Under Outputs, select the output parameter whose output you want to map. Click the curly brackets to see a list of the output parameters available from the script.</li> <li>3. Under Field to fill, select the field that you want to populate with the output.</li> <li>4. Click Save.</li> </ol> </li> <li>Advanced: Includes the following fields. <ul style="list-style-type: none"> <li>Using: Choose which integration instance will execute the command, or leave empty to use all integration instances.</li> <li>Extend context: Append the extracted results of the action to the context. For example, <code>"newContextKey1=path1::newContextKey2=path2"</code> returns <code>"[path1:'aaa',path2:'bbb', newContextKey1:'aaa',newContextKey2:'bbb']"</code></li> <li>Ignore outputs: If set to true, will not store outputs into the context (besides the extended outputs).</li> <li>Execution timeout (seconds): Sets the command execution timeout in seconds.</li> <li>Indicator Extraction mode: Choose when to extract indicators: <ul style="list-style-type: none"> <li>None: Do not perform indicator extraction</li> <li>Inline: Before other playbook tasks</li> <li>Out of band: While other tasks are running</li> </ul> </li> <li>Mark results as note</li> <li>Mark results as evidence</li> <li>Run without a worker</li> <li>Skip this branch if this script/playbook is unavailable</li> <li>Quiet Mode: When in quiet mode, tasks do not display inputs and outputs or extract indicators. Errors and warnings are still documented. You can turn quiet mode on or off at the task or playbook level.</li> </ul> </li> <li>Details: Includes the following fields. <ul style="list-style-type: none"> <li>Tag the result with: Add a tag to the task result. You can use the tag to filter entries in the War Room.</li> <li>Task description (Markdown supported): Describe what this task does. You can enter objects from the context data in the description. For example, in a communication task, you can use the recipient's email address. The value for the object is based on what appears in the context every time the task runs.</li> </ul> </li> <li>On Error: Includes the following fields.</li> </ul>



Field / Tab	Settings
	<ul style="list-style-type: none"> <li>◦ Number of retries: How many times the task should retry running if there is an error. Default is 0.</li> <li>◦ Retry interval (seconds): How long to wait between retries. Default is 30 seconds.</li> </ul> <p>The maximum retry interval is 800 seconds (13.3 minutes). If you enter a value greater than 800 seconds, the retry interval will be limited to 800 seconds.</p> <ul style="list-style-type: none"> <li>◦ Error handling: How the task should behave if there is an error. Options are: <ul style="list-style-type: none"> <li>▪ Stop</li> <li>▪ Continue</li> <li>▪ Continue on error path(s)</li> </ul> </li> </ul> <p>This option configures the task to handle potential errors that may occur when executing the current task's script.</p>
Manual task settings tab	<ul style="list-style-type: none"> <li>• Default assignee: Assign an owner to this task.</li> <li>• Only the assignee can complete the task: Stop the playbook from proceeding until the task assignee completes the task. By default, in addition to the task assignee, the default administrator can also complete the blocked task. You can also block tasks until a user with an external email address completes the task.</li> <li>• Task SLA: Set the SLA in granularity of weeks, days, hours, and minutes.</li> <li>• Set task Reminder at: Set a reminder for the task in the granularity of weeks, days, hours, and minutes.</li> </ul>
Advanced tab	<p>Quiet Mode: Determines whether this task uses the playbook default setting for quiet mode. When in quiet mode, tasks do not display inputs and outputs or extract indicators. Errors and warnings are still documented. You can turn quiet mode on or off at the task or playbook level.</p>
Details tab	<ul style="list-style-type: none"> <li>• Tag the result with: Add a tag to the task result. You can use the tag to filter entries in the War Room.</li> <li>• Task description (Markdown supported): Provide a description of what this task does. You can enter objects from the context data in the description. For example, in a communication task, you can use the recipient's email address. The value for the object is based on what appears in the context every time the task runs.</li> </ul>

5. Click Save.

The task is added in the playbook editor.

If you selected a system script in the settings, the task logo indicates Builtin.

6. Connect the tasks you've added in their logical order by dragging and dropping a wire from one task to another.

7. Save the playbook.

#### Create a conditional task

##### Abstract

#### Create a Conditional task in a playbook.

Conditional tasks are used for determining different paths for your playbook. For example, in a playbook for handling phishing emails, a conditional task can be used to check if an email contains suspicious attachments. If the attachment is identified as malicious, the playbook can automatically quarantine the email; otherwise, it can proceed to manual review by a security analyst.

##### Conditional task types

You can create different types of conditional tasks.



- Built-in: Creates a logical statement using an entity from within the playbook. For example, in an access investigation playbook, you can determine that if the Asset ID of the person whose account was being accessed exists in a VIP list, set the issue severity to High. Otherwise, proceed as normal.
- Manual: Creates a conditional task that must be manually resolved. For example, a security analyst is prompted to review and validate a suspicious file. The playbook task might involve instructions for the analyst to analyze the file, determine if it is malicious, and provide feedback or take specific actions based on their assessment.
- Ask: Creates a single-question survey communication task, the answer to which determines how a playbook proceeds. For more details about ask tasks, see Create a communication task.
- Choose script: Creates a conditional task based on the result of a script. For example, check if an IP address is internal or external using the `IsIPInRanges` script. When using a script, the inputs and outputs are generated by the automation script.

#### How to create a conditional task

1. From the Task Library pane, click the task you want, for example Blank Task.
2. In the Task Details pane, select the Conditional Task Type.
3. In the Task Name field, type a meaningful name for the task that corresponds to the data you are collecting.
4. Select the relevant conditional task option. Some field configurations are required, and some are optional.

#### Built-in

- Condition: Define one or more logical conditions for the task.
- Details: Includes the following fields.
  - Tag the result with: Add a tag to the task result. You can use the tag to filter entries in the War Room.
  - Task description (Markdown supported): Provide a description of what this task does. You can enter objects from the context data in the description. For example, in a communication task, you can use the recipient's email address. The value for the object is based on what appears in the context every time the task runs.
- Advanced: Determines whether this task uses the playbook default setting for Quiet Mode. When in Quiet Mode, tasks do not display inputs and outputs or extract indicators. Errors and warnings are still documented. You can turn Quiet Mode on or off at the task or playbook level.
- On Error: Includes the following fields.
  - Number of retries: How many times the task should retry running if there is an error. Default is 0.
  - Retry interval (seconds): How long to wait between retries. Default is 30 seconds.

#### Manual

- Manual task settings: Includes the following fields.
  - Default assignee: Assign an owner to this task.
  - Only the assignee can complete the task: Stop the playbook from proceeding until the task assignee completes the task. By default, in addition to the task assignee, the default administrator can also complete the blocked task. You can also block tasks until a user with an external email address completes the task.
  - Task SLA: Set the SLA in granularity of weeks, days, hours, and minutes.
  - Set task Reminder at: Set a reminder for the task in granularity of weeks, days, hours, and minutes.
- Advanced: Determines whether this task uses the playbook default setting for Quiet Mode. When in Quiet Mode, tasks do not display inputs and outputs or extract indicators. Errors and warnings are still documented. You can turn Quiet Mode on or off at the task or playbook level.
- Details: Includes the following fields.
  - Tag the result with: Add a tag to the task result. You can use the tag to filter entries in the War Room.
  - Task description (Markdown supported): Provide a description of what this task does. You can enter objects from the context data in the description. For example, in a communication task, you can use the recipient's email address. The value for the object is based on what appears in the context every time the task runs.

#### Ask



- Message: Includes the following fields.
  - Ask by: The method for sending the message and survey. Options are:
    - Task (can always be completed directly in the Workplan)
    - Generated link (appears in the context data)
    - Email
  - To: The message and survey recipients. You can define by:
    - Selecting from a predefined drop down list.
    - Manually typing email addresses for users and/or external users.
    - Clicking the context icon to define recipients from a context data source.
  - CC of the email: A CC email address.
  - Subject of the email: The message subject that displays to message recipients. You can write the survey question in the subject field or in the message body field.
  - Message body: The text that displays in the body of the message. This field is optional, but if you don't write the survey question in the subject field, include it in the message body. This is a long-text field.
  - Reply options: Reply options are sent via the selected channels as options for an answer.
  - Require users to authenticate: Enable this option to have your SAML or AD authenticate the recipient before allowing them to answer. You must first set up an authentication integration instance and check Use this instance for external users authentication only in the integration instance settings.
- Timing: Includes the following fields.
  - Retry interval (minutes): Determines the wait time between each execution of a command. For example, the frequency (in minutes) that a message and survey are resent to recipients before the response is received.
  - Number of retries: Determines how many times a command attempts to run before generating an error. For example, the maximum number of times a message is sent. If a reply is received, no additional retry messages will be sent.
  - Task SLA: Set the SLA in granularity of weeks, days, and hours.
  - Set task Reminder at: Set a task reminder in the granularity of weeks, days, and hours.
  - Complete automatically if SLA passed without a reply: Select this checkbox to complete the task if the SLA is breached before a reply is received. You can select yes or no.
- Advanced: Includes the following fields.
  - Using: Choose which integration instance will execute the command, or leave empty to use all integration instances.
  - Extend context: Append the extracted results of the action to the context. For example, "newContextKey1=path1::newContextKey2=path2" returns "[path1:'aaa',path2: 'bbb', newContexKey1: 'aaa',newContextKey2:'bbb']"
  - Ignore outputs: If set to true, will not store outputs into the context (besides the extended outputs).
  - Execution timeout (seconds): Sets the command execution timeout in seconds.
  - Indicator Extraction mode: Choose when to extract indicators:
    - None: Do not perform indicator extraction
    - Inline: Before other playbook tasks
    - Out of band: While other tasks are running
  - Mark results as note
  - Mark results as evidence
  - Run without a worker
  - Skip this branch if this script/playbook is unavailable
  - Quiet Mode: When in quiet mode, tasks do not display inputs and outputs or extract indicators. Errors and warnings are still documented. You can turn quiet mode on or off at the task or playbook level.
- Details: Includes the following fields.



- Tag the result with: Add a tag to the task result. You can use the tag to filter entries in the War Room.
- Task description (Markdown supported): Describe what this task does. You can enter objects from the context data in the description. For example, in a communication task, you can use the recipient's email address. The value for the object is based on what appears in the context every time the task runs.

Choose script

From a drop-down list, select a script for the playbook to run. In the following tabs, you can set:



- Inputs: Each script has its own set of input arguments (or none). You can set each argument to a specific value (by typing directly on the line under the argument name), or you can click the curly brackets to define a source field to populate the argument.

- Outputs: Each script has its own set of output arguments (or none).

- Mapping:

Map the output from a playbook task directly to an issue field.

The value for an output key populates the specified field per issue. This is a good alternative to using a task with a `setIssue` command.

**NOTE:**

The output value is dynamic and is derived from the context at the time that the task is processed. As a result, parallel tasks that are based on the same output may return inconsistent results.

1. In the Mapping tab, click Add custom output mapping.
  2. Under Outputs, select the output parameter whose output you want to map. Click the curly brackets to see a list of the output parameters available from the automation.
  3. Under Field to fill, select the field that you want to populate with the output.
  4. Click Save.
- Advanced: Includes the following fields.
    - Using: Choose which integration instance will execute the command, or leave empty to use all integration instances.
    - Extend context: Append the extracted results of the action to the context. For example, "newContextKey1=path1::newContextKey2=path2" returns "[path1:'aaa',path2: 'bbb', newContexKey1: 'aaa',newContextKey2:'bbb']"
    - Ignore outputs: If set to true, will not store outputs into the context (besides the extended outputs).
    - Execution timeout (seconds): Sets the command execution timeout in seconds.
    - Indicator Extraction mode: Choose when to extract indicators:
      - None: Do not perform indicator extraction
      - Inline: Before other playbook tasks
      - Out of band: While other tasks are running
    - Mark results as note
    - Mark results as evidence
    - Run without a worker
    - Skip this branch if this script/playbook is unavailable
    - Quiet Mode: When in quiet mode, tasks do not display inputs and outputs or extract indicators. Errors and warnings are still documented. You can turn quiet mode on or off at the task or playbook level.
  - Details: Includes the following fields.
    - Tag the result with: Add a tag to the task result. You can use the tag to filter entries in the War Room.
    - Task description (Markdown supported): Provide a description of what this task does. You can enter objects from the context data in the description. For example, in a communication task, you can use the recipient's email address. The value for the object is based on what appears in the context every time the task runs.
  - On Error: Includes the following fields.
    - Number of retries: How many times the task should retry running if there is an error. Default is 0.
    - Retry interval (seconds): How long to wait between retries. Default is 30 seconds.
    - Error handling: How the task should behave if there is an error. Options are:
      - Stop
      - Continue
      - Continue on error path(s)

This option configures the task to handle potential errors that may occur when executing the current task's script.

5. Click Save.



The task is added in the playbook editor.

If you selected a system script in the settings, the task logo indicates Builtin.

6. Connect the tasks you've added in their logical order by dragging and dropping a wire from one task to another.

7. Save the playbook.

## Create a communication task

### Abstract

Communication tasks in playbooks enable you to send surveys and collect data. Ask task, data collection task.

Communication tasks enable you to send surveys to users, both internal and external, to collect data for an issue. The collected data can be used for issue analysis, and also as input for subsequent playbook tasks. For example, you can send a scheduled survey requesting analysts to send specific issue updates or send a single (stand-alone) question survey to determine how an issue was handled.

There are two types of communication tasks:

- **Ask tasks:** A conditional task that sends a single question survey. The answer is used to determine how the playbook proceeds.
- **Data collection tasks:** A data collection task sends a survey of one or more questions. The answers are recorded in context data and can be used as input for subsequent tasks.

### About ask tasks

An ask task is a type of conditional task that sends a single question survey, the answer to which determines how a playbook proceeds. If you send the survey to multiple users, the first answer received is used, and subsequent responses are disregarded. For more information about ask task settings, see Create a conditional task.

Because this is a conditional task, you need to create a condition for each of the answers. For example, if the survey answers include, **Yes**, **No**, and **Maybe**, there should be a corresponding condition (path) in the playbook for each of these answers.

Users interact with the survey directly from the message, meaning the question appears in the message and they click an answer from the message.

The survey question and the first response is recorded in the issue context data. This enables you to use this response as the input for subsequent playbook tasks.

For all ask conditional tasks, a link is generated for each possible answer the recipient can select. If the survey is sent to more than one user, a unique link is created for each possible answer for each individual recipient. These links are visible in the context data of the issue's Work Plan. The links appear under Ask.Links in the context data.

### Example 29. Send a survey

In this example, the message and survey will be sent to recipients every hour for six hours, until a reply is received (it is repeated every 60 minutes, 6 times). The SLA is six hours. If the SLA is breached, the playbook will proceed according to the Yes condition.

The screenshot shows the 'TASK DETAILS' dialog box. At the top, there are four radio button options: Standard (unchecked), Conditional (checked), Data Collection (unchecked), and Section Header (unchecked). Below this, there is a question field labeled 'Was the Alert escalated?' with a help icon (info symbol inside a circle) to its right. Underneath the question, there are four automation options: Built-in (unchecked), Manual (unchecked), Ask (checked), and Choose automation (unchecked). The 'Timing' tab is selected, showing a 'Retry interval (minutes)' of 60 and a 'Number of retries' of 6. The 'Task SLA' section shows '0 Weeks 0 Days 6 Hours'. Below the SLA, there is a link 'Set task Reminder at'. At the bottom, there is a note: 'Complete automatically if SLA passed without a reply. Use option:' followed by a dropdown menu with 'Yes' selected.



### Example 30. Send email to users

In this example, a message and survey are sent by email to all users with the Analyst role. We are not including a message body because the message subject is the survey question we want recipients to answer. There are three reply options, Yes, No, and Not sure. In the playbook, we will only add conditions for the Yes and No replies. We require recipient authentication, which first involves setting up authentication.

The screenshot shows the 'TASK DETAILS' configuration screen. The task type is set to 'Conditional'. The title of the task is 'email survey'. The communication method is selected as 'Ask'. The 'Message' tab is active. Under 'Ask by', the 'Email' option is checked. The 'To' field contains 'Analyst'. The 'Subject of the email' field contains 'Was the incident escalated?'. The 'Message body' field is empty and labeled 'Text'. Under 'Reply Options', there are two options: 'Yes' and 'No'. A link '+ Add reply option' is available. A checkbox 'Require users to authenticate' is checked. At the bottom, there is a link 'Create a data collection task'.

#### Create a data collection task

The data collection task is a multi-question survey (form) that survey recipients access from a link in the message. Users do not need to log in to access the survey, which is located on a separate site.

All responses are collected and recorded in the issue context data, whether you receive responses from a single user or multiple users. This enables you to use the survey questions and answers as input for subsequent playbook tasks. If responses are received from multiple users, data for multi-select fields and grid fields are aggregated. For all other field types, the response received most recently will override previous responses as it displays in the field. All responses are always available in the context data.

For all data collection tasks, a single link is generated for each recipient of the survey. These links are visible in the context data of the issue's Work Plan. The links appear in the context data under the Links section of that survey.

You can include the following types of questions in the survey.

- Stand alone questions. These are presented to users directly in the message, and from which users answer directly in the message (not an external survey).
- Field-based questions. These are based on a specific issue field (either system or custom), for example, an Asset ID field. The response (data) received for these fields automatically populates the field for this issue. For single-select field based questions, the default option is taken from the field's defined default.



## How to create a Data Collection task

1. From the Task Library pane, click the task you want, for example Blank Task.
2. In the Task Details pane, select the Data Collection Task Type.
3. Enter a meaningful name in the Task Name field for the task that corresponds to the data you are collecting.
4. Select the communication options you want to use to collect the data.

### Tabs and configuration fields

Tab	Configuration Fields In The Tab
Message	<ul style="list-style-type: none"><li>• Ask by: The method for sending the message and survey. Options are:<ul style="list-style-type: none"><li>◦ Task (can always be completed directly in the Workplan)</li><li>◦ Generated link (appears in the context data): A link to the data collection survey is available in the context data of the task.</li><li>◦ Email: If you select this option, enter below the subject and message of the email and the email addresses of the users who should receive this message or survey.</li></ul></li><li>• To: The message and survey recipients. You can define by:<ul style="list-style-type: none"><li>◦ Selecting from a predefined drop down list.</li><li>◦ Manually typing email addresses for users and/or external users.</li><li>◦ Clicking the context icon to define recipients from a context data source.</li></ul></li><li>• CC of the email: A CC email address.</li><li>• Subject of the email: The message subject that displays to message recipients. You can write the survey question in the subject field or in the message body field.</li><li>• Message body: The message question body to be used in the notification sent to the given users along with the reply options.</li><li>• Require users to authenticate: Enable this option to have your SAML or AD authenticate the recipient before allowing them to answer. You must first set up an authentication integration instance and check Use this instance for external users authentication only in the integration instance settings.</li></ul>
Questions	<ul style="list-style-type: none"><li>• Web Survey Title: The title displayed for the web survey.</li><li>• Short Description: A description displayed above the questions on the web survey. Click Preview to see how it displays.</li><li>• Question: A question to ask recipients.</li><li>• Answer Type: The field type for the answer field. Options are:<ul style="list-style-type: none"><li>◦ Short text</li><li>◦ Long text</li><li>◦ Number</li><li>◦ Single Select (requires you to define a reply option)</li><li>◦ Multi select/Array (requires you to define a reply option)</li><li>◦ Date picker</li><li>◦ Attachments</li></ul></li><li>• Mandatory: If this checkbox is selected for a question, survey recipients will not be able to submit the survey until they answer this question.</li><li>• Help Message: The message that displays when users hover over the question mark help button for the survey question.</li><li>• Placeholder: A sample value displayed until a real value is entered.</li></ul> <p><b>NOTE:</b></p> <p>You can drag questions to rearrange the order in which they display in the survey.</p>



Tab	Configuration Fields In The Tab
Timing	<ul style="list-style-type: none"> <li>Retry interval (minutes): Determines the wait time between each execution of a command. For example, the frequency (in minutes) that a message and survey are resent to recipients before the response is received.</li> <li>Number of retries: Determines how many times a command attempts to run before generating an error. For example, the maximum number of times a message is sent. If a reply is received, no additional retry messages will be sent.</li> </ul> <p><b>NOTE:</b></p> <p>Retries are not supported for data collection tasks that have errors sending emails (indicated by a server timeout). This is because retries only work on automation execution failures, not on email delivery issues.</p> <ul style="list-style-type: none"> <li>Task SLA: Set the SLA in granularity of weeks, days, and hours.</li> <li>Set task Reminder at: Set a task reminder in granularity of weeks, days, and hours.</li> <li>Complete automatically if: <ul style="list-style-type: none"> <li>Reached task SLA (with or without a reply): This option is grayed out.</li> <li>Received &lt;enter a number&gt; reply</li> </ul> </li> </ul>
Details	<ul style="list-style-type: none"> <li>Tag the result with: Add a tag to the task result. You can use the tag to filter entries in the War Room.</li> <li>Task description (Markdown supported): Describe what this task does. You can enter objects from the context data in the description. For example, in a communication task, you can use the recipient's email address. The value for the object is based on what appears in the context every time the task runs.</li> </ul>
Advanced	<ul style="list-style-type: none"> <li>Using: Choose which integration instance will execute the command, or leave empty to use all integration instances.</li> <li>Extend context: Append the extracted results of the action to the context. For example, "newContextKey1=path1::newContextKey2=path2" returns "[path1:'aaa',path2: 'bbb', newContextKey1: 'aaa',newContextKey2:'bbb']"</li> <li>Ignore outputs: If set to true, will not store outputs into the context (besides the extended outputs).</li> <li>Execution timeout (seconds): Sets the command execution timeout in seconds.</li> <li>Indicator Extraction mode: Choose when to extract indicators: <ul style="list-style-type: none"> <li>None: Do not perform indicator extraction</li> <li>Inline: Before other playbook tasks</li> <li>Out of band: While other tasks are running</li> </ul> </li> <li>Mark results as note</li> <li>Mark results as evidence</li> <li>Run without a worker</li> <li>Skip this branch if this script/playbook is unavailable</li> <li>Quiet Mode: When in quiet mode, tasks do not display inputs and outputs or extract indicators. Errors and warnings are still documented. You can turn quiet mode on or off at the task or playbook level.</li> </ul>

5. (Optional) To customize the look and feel of your email message, click Preview.

You can determine the color scheme and how the text in the message header and body appear, as well as the appearance and text of the button the user clicks to submit the survey.

If you configured a custom logo in server settings, it will appear in the preview.

**NOTE:**

When customizing HTML for data collection emails, do not apply CSS styles directly to the `<body>` tag. Cortex AgentIX injects your HTML as a fragment into an existing email template and removes the `<body>` tag to ensure valid HTML structure. Any styles applied to the `<body>` tag will be lost. To ensure your formatting renders correctly, wrap your content in a container element such as a `<div>` or `<span>` and apply your styles to that container.

```
<body>
<div style="font-family: sans-serif;">Content</div>
```



</body>

6. Click Save.

The task is added in the playbook editor.

A user icon (  ) indicates the task requires manual inputs.

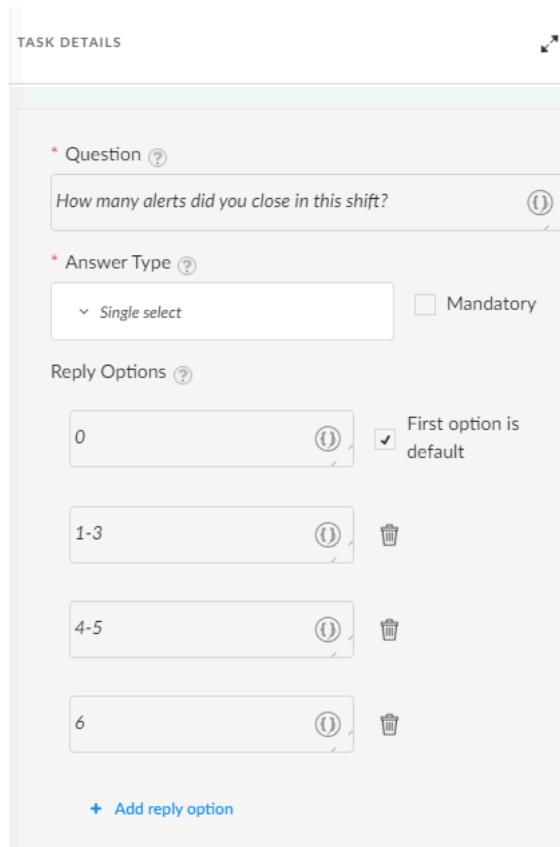
7. Connect the tasks you've added in their logical order by dragging and dropping a wire from one task to another.

8. Save the playbook.

Data collection task examples

Stand-alone question with a single-select answer

In this example, we create a stand-alone question, with a single-select answer. This question is not mandatory. If we selected the First option is default checkbox, the reply option "0" is the default value in the answer field.



The screenshot shows the 'TASK DETAILS' interface for a stand-alone question. The 'Question' field contains the text 'How many alerts did you close in this shift?'. The 'Answer Type' dropdown is set to 'Single select'. A 'Mandatory' checkbox is unchecked. Under 'Reply Options', there are four entries: '0' (selected as the first option), '1-3', '4-5', and '6'. A checked checkbox next to '0' indicates it is the 'First option is default'. Below the options is a '+ Add reply option' button.

Field-based using a custom field

In this example, we create a question based on a custom issue field that is marked as mandatory. You can add a question based on a field. To add a field, click the Add Question based on field.



QUESTION 1: PLEASE DETAIL THE CLOSED ALERTS

FOR YOUR SHIFT

\* Question [?](#) [Detach from field](#)

Please detail the Closed Alerts for your shift [\(i\)](#)

\* Field associated with this question [?](#)

Analyst Status Report: Closed Alerts  Mandatory

Help Message [?](#)

Placeholder [?](#)

[+ Add Question](#) [+ Add Question based on field](#) [Top of page](#)

#### Configure communication task authentication

When sending a form in a communication task, you can configure user authentication to ensure only authorized users gain access to the form.

The authorized users are usually external users not in Cortex AgentiX, and they will not be able to access anything else in Cortex AgentiX.

#### Set up playbook communication task authentication

1. Set up your SSO if it is not already configured. See [Authenticate users using SSO](#) for more details.
2. In the Task details of your playbook communication task, check [Require users to authenticate](#) to have your SAML or AD authenticate the recipient before allowing them access to the form.

TASK DETAILS

Standard  Conditional  Data Collection  Section Header

#1 Untitled Task [\(i\)](#)

Attributes: Quiet Mode

[Message](#) [Questions](#) [Timing](#) [Details](#) [Advanced](#)

Ask by [Preview](#)

Select communication options:

- Task (can always be completed directly in the workplan)
- Generated link (appears in the context data)
- Email

\* To

Select from predefined values or add your own [\(i\)](#)

CC of the email

Select from predefined values or add your own [\(i\)](#)

\* Subject of the email

Message body [Text](#)

[\(i\)](#)

Link to web form will be placed automatically at the bottom of your message

Require users to authenticate [\(i\)](#)



Configure NGINX as a reverse proxy to access data collection links in emails

If you are using NGINX as a reverse proxy with SSL termination, configure the NGINX configuration file to enable accessing data collection links in emails.

1. Navigate to `/etc/nginx/sites-available/` and open the NGINX configuration file.
2. Update the file with the following configurations:

```
server {  
    listen 443 ssl;  
    server_name <PROXY DOMAIN>;  
  
    ssl_certificate <path to CRT file>;  
    ssl_certificate_key <path to KEY file>;  
    ssl_protocols TLSv1.2 TLSv1.3;  
    ssl_ciphers 'ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384';  
    ssl_prefer_server_ciphers on;  
  
    location / {  
        proxy_pass https://<XSOAR DOMAIN>;  
        proxy_cookie_domain <XSOAR DOMAIN> <PROXY DOMAIN>;  
        proxy_pass_header Set-Cookie;  
        proxy_set_header X-Real-IP $remote_addr;  
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;  
        proxy_set_header X-Forwarded-Proto $scheme;  
    }  
}
```

3.8.4.5.3.5 | [Create a section header](#)

## Abstract

Section headers are used to manage the flow of your playbook and help you organize your tasks efficiently.

Section headers are used to manage the flow of your playbook and help you organize your tasks efficiently. You create a section header to group a number of related tasks.

1. From the Task Library pane, click Header or Blank Task.
2. In the Task Details pane, for Task Type, select the Section Header icon.
3. Enter a meaningful name in the Task Name field for the section header.
4. In the Details tab, configure the following.
  - Tag the result with: Add a tag to the task result. You can use the tag to filter entries in the War Room.
  - Sub Section: If selected, this section becomes a subsection of the parent section above it, and it collapses when its parent section collapses.
  - Task description (Markdown supported): Provide a description of what this task does. In the Playbooks page, click  on the section header to display the description.
5. Click Save.

3.8.4.5.3.6 | [Configure script error handling in a playbook](#)

## Abstract

When defining a task, you can decide if the playbook continues, stops, or continues on an error path.

You can determine how the playbook behaves if there are script errors during execution.

When defining a standard task that uses a script or a conditional task that uses an script, you can define how a playbook task continues by selecting one of the following options:



- Stop: The playbook stops, if the task errors during execution. For example, if the task requires a manual review, you may want the playbook to stop until completion.
- Continue: The playbook continues to execute if the task errors. For example, the playbook task requires EWS, but EWS is not required for the playbook to proceed.
- Continue on error path: If a task errors, the playbook continues on an error path.

The error path may be useful if you want to take action on an error, like clean-up, retry, etc. You may also want to handle errors in different ways. For example, in case of a quota expired error you may want to retry in 1 minute, but if you receive an internal error 500, you may want to stop the playbook.

You may want to create a separate path when an analyst manually reviews the issue and research is needed outside Cortex AgentX. Once an analysis is complete, you can add a task to consider escalating to a customer and, if so, generate a report which can be attached to a ticket system such as Jira or ServiceNow.

Instead of a playbook waiting on manual input, which displays an error state, such as missing an argument in a script, you can add a separate path for these kinds of issues.

#### **NOTE:**

Use the **GetErrorsFromEntry** script (part of the Common Scripts Pack) to check whether the given entry returns an error and returns an error message. For example, when using the script in a playbook, you can fetch the error message from a given task, such as a runtime error. You can then add a step in the playbook flow to send those error messages to the relevant stakeholder through Slack, email, opening a Jira ticket, etc.

When errors are created, they are added to context under `task.id.error`.

How to set up error handling in your playbook

1. In a playbook, edit a task or create a task from the Task Library.
2. In the Task Details pane, set the Task Type to either Standard or Conditional.

#### **NOTE:**

You can set up script error handling only when running a script in a Standard task or a Conditional task. For more information about error handling settings for these tasks, see [Create a standard task](#) or [Create a conditional task](#).

**Built-in, Manual, and Ask Conditional tasks have On Error settings for number of retries and retry interval, but not Error Handling.**

3. Select a script.
4. Click the On Error tab.
5. In the Number of retries field, type the number of times the tasks attempts to run before generating an error.
6. In the Retry Interval (seconds) field, type the wait time between retrying the task.
7. In the Error Handling field, select one of the following:

- Stop
- Continue
- Continue on error path(s)

8. Click Save.
9. When adding a connector from this task to the next task, a dialog box appears which enables you to select one of the following paths:
  - Standard Path: When adding a task to this path, it executes without any exceptions.  
If you select the Standard Path, the task continues on this path and executes without exceptions.
  - Error Path: When adding a task to this path, it executes where the source task errors during execution.  
If you select Error path, if the task errors, the playbook continues with this path.

#### 3.8.4.5.4 | Task 4. Add custom playbook features

##### Abstract

Use an out-of-the-box playbook, create a new playbook, or customize an existing one based on your organization's needs.

You can customize your playbook to do the following.



Custom Action	Description
Configure a sub-playbook loop	Automate the execution of a series of actions in a sub-playbook loop to enable handling repetitive tasks efficiently, increasing workflow productivity and consistency.
Filter and transform data	Filters extract relevant data to help focus on relevant information and discard irrelevant or unnecessary data. Transformers take one value and transform or render it to another value or format.
Use scripts	Perform specific automated actions using commands which are also used in playbook tasks and in the War Room. Configure script error handling.
Extract indicators	Extract indicators from issue fields and enrich them using commands and scripts defined for the indicator type.
Extended context	Save additional data from the raw response of commands that return data.
Update issue fields with playbook tasks	Use the setIssue script to set and update all system issue fields.
Create an automation rule	Create conditions so if an issue with specific characteristics is created, a suitable response is issued via a playbook.
Use playbook polling	Configure a playbook to stop and wait for a process to complete on a third-party product, and continue when it is done.

#### 3.8.4.5 | Task 5. Test and debug the playbook

##### Abstract

Use an out-of-the-box playbook, create a new playbook, or customize an existing one based on your organization's needs.

The debugger provides a test environment where you can make changes to data and playbook logic and view the results in real-time to test and troubleshoot playbooks. You can see exactly what is written to the context at each step and which indicators are extracted.

For more information, see [Test your playbook](#).

#### 3.8.4.5.6 | Task 6. Manage playbook content

##### Abstract

Use an out-of-the-box playbook, create a new playbook, or customize an existing one based on your organization's needs.

Manage playbook content by saving versions of your playbook in Cortex AgentiX to maintain version history.

Cortex AgentiX also manages playbook editing conflicts by only allowing one user at a time to edit a playbook and locking it for other users to edit. It automatically clears the playbook edit lock when the user currently editing the playbook saves the playbook, closes the editor, logs out, or when their session expires. In addition, you can grant a permission to designated users so they can manually unlock playbooks from the Playbooks page or editor.

For more details, see [Manage playbook content](#).

#### 3.8.4.6 | Customize your playbook

##### Abstract

Customize your playbook to extract indicators, extend context, update issue fields, filter and transform data, run scripts, and perform triggered actions, and sub-playbook loops.



Customizing a playbook helps you automate tasks to match your needs, making workflows more efficient, accurate, and easier to integrate with your existing processes.

#### 3.8.4.6.1 | Configure a sub-playbook loop

##### Abstract

Configure a sub-playbook to run in a loop.

Looping uses sub-playbooks to create loops within the main playbook. When running the loop, the values are calculated based on the context data for the sub-playbook and not the main playbook.

##### **NOTE:**

Consider the following when adding a loop:

- The maximum number of loops (default is 100). A high number of loops or a high wait time combined with a large number of issues may affect performance.
- Periodically check looping conditions to ensure they are still valid for the data set.
- If you want a sub playbook task to loop over an array passed into its input, you need to configure a loop. Otherwise it takes in the whole array and runs once.

##### How to create a sub-playbook loop

1. In the Playbooks page, select the parent playbook that contains the sub-playbook task you want to run in a loop.

2. Right click and select Edit.

If the playbook is installed from a content pack, you need to duplicate or detach the playbook before editing.

3. Click the sub-playbook for which you want to create the loop.

4. In the Task Details pane, click the Loop tab.

5. Click one of the following options to define loop settings:

- None: (Default) The sub-playbook does not loop.
- Built-in: Use built-in functions to define loop settings:

Option	Description
Exit when	Enables you to define when to exit the loop. Click {} and expand the source category. Hover over the required source and click <b>Filter &amp; Transform</b> to the left of the source to manipulate the data.
Equals (String)	Select the operator to define how the values should be evaluated.
Max iterations	The number of times the loop should run. <b>TIP:</b> Balance between the number of iterations and the interval so you do not overload the server.
Sleep	The number of seconds to wait between iterations. recommends that you balance between the number of iterations and the number of seconds to wait between iterations so you don't overload the server.

- For each input: Runs the sub-playbook based on defined inputs. Enter the number of seconds to wait between iterations.
- Choose Loop automation: Select the automation from the drop-down list to define when to exit the loop. The parameters that appear are applicable to the selected automation.

6. To save the changes, click OK.

Example: Exit looping after running for all inputs

In the parent playbook (the task that contains the sub-playbook), you can configure to exit a loop running the sub-playbook automatically when the last item in the sub-playbook input is executed.



- If the input is a single item, the sub-playbook runs once, but if the input is a list of items (such as a list of issue IDs), the sub-playbook runs as many times as there are items in the list. Each iteration of the sub-playbook uses the next item in the list as the input.
- If there are multiple input lists with the same amount of items, the sub-playbook runs once for each set of inputs.
- If there are multiple input lists with different amounts of items, the sub-playbook runs the first set of inputs, followed by the second, third, and so on, until the end.

For example:

Input	Value
Input x	1,2,3,4
Input y	a,b,c,d
Input z	9

The first loop: 1, a, 9

The second loop: 2, b, 9

The third loop: 3, c, 9

The fourth loop: 4, d, 9

#### 3.8.4.6.2 | Filter and transform data

##### Abstract

Use filters and transformers to manipulate data. Use filters and transformers in playbook tasks or when mapping an instance.

In Cortex AgentiX, data is extracted and collected from various sources, such as playbook tasks, command results, and fetched issues, and presented in JSON format. The data can be manipulated by using filters and transformers.

##### Filters

Filters enable you to extract relevant data which you can use elsewhere in Cortex AgentiX. For example, if an issue has several files with varying file types and extensions, you can filter the files by file extension or file type, and use the filtered files in a detonation playbook. You can filter as many objects as required. Cortex AgentiX automatically calculates the context root to which to filter. You can change the context root as necessary.

##### CAUTION:

You can change the context data root to filter, but it is not recommended to select a different root, as it affects the filter results. The drop-down list displays the filter root for backward compatibility.

##### Transformers

Transformers modify or format data to make it suitable for further processing or presentation. For example, you can convert a date in non-Unix format to Unix format. Another example is applying the `count` transformer, which renders the number of elements. When you have more than one transformer, they apply in the order that they appear. You can reorder them using click-and-drag.

##### Add filters and transformers in a playbook task

1. Create or edit a playbook task.
2. In the field you want to add a filter or transformer (for example, inputs or outputs), click the curly brackets and then select Filters and Transformers.
3. In the Get field, type or select data you want to filter or transform. For example, `EWS.Items.Name`.
4. (Optional) To filter the data, do the following.
  - a. In the Filter section, click Add filter.  
When adding a filter, the context root to filter is automatically populated.
  - b. Select the data you want to filter.
  - c. Select the filter operators.



d. Add the value.

e. Click the checkbox to save the filter.

5. (Optional) To apply transformers to the field, click Add transformer.

a. Click the transformer and select the relevant transformer.

By default, the transformer is set to `To upper case(String)`. Click it to pick a different transformer, for example to change the date format for when issues occurred.

b. Select the transformer operators.

c. Click the tick box to save.

6. (Optional) To test the filter or transformation click Test and select the investigation or add it manually.

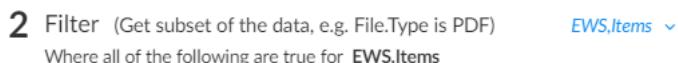
Example: Filter items with an EXE extension

In this example, we want to filter all EWS Item names that have the extension `exe`.



1. From the Filters & transformers window, in the Get field, type `EWS.Items.Name` to extract all Item names in EWS.

The context root to filter is `EWS.Items`.

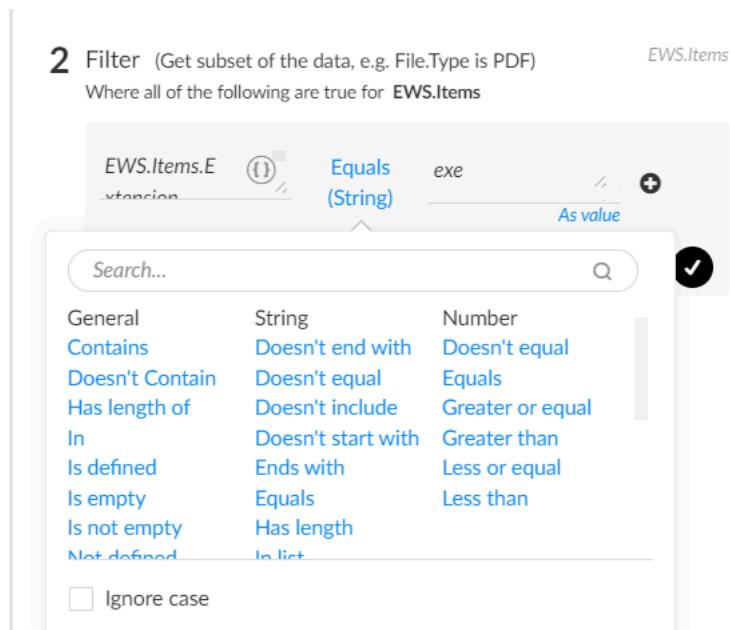


2. In the Filter section, click Add filter.

3. In the left-hand side, add `Extension` to the filter.

4. Select Equals (String)  ignore case.

5. In the right-hand side add `exe`.



6. Click the tick box to save the filter.

7. Click Test.



You should see Item names are filtered with the extension **.exe**.

Example (advanced): Filter hostname for the last resolved time

In this example, we want to see the **LastResolved** time only from the **demisto.com** host name.

This is part of the data where we want to filter:

```
{
    "IP": [
        {
            "Address": "192.168.10.96",
            "AutoFocus": {
                "Resolutions": [
                    {
                        "Hostname": "79463wwfqq,dattolocal.net",
                        "LastResolved": "2022-08-02 04:01:02"
                    },
                    {
                        "Hostname": "demisto.com",
                        "LastResolved": "2022-09-10 09:47:17"
                    },
                    {
                        "Hostname": "securesense.call4pchelp.com",
                        "LastResolved": "2022-04-22 11:49:06"
                    }
                ]
            }
        },
        {
            "Address": "192.168.10.96",
            "AutoFocus": {
                "Resolutions": [
                    {
                        "Hostname": "79463wwfqq,dattolocal.net",
                        "LastResolved": "2022-08-02 04:01:02"
                    },
                    {
                        "Hostname": "demisto.com",
                        "LastResolved": "2022-09-10 09:47:17"
                    },
                    {
                        "Hostname": "securesense.call4pchelp.com",
                        "LastResolved": "2022-04-22 11:49:06"
                    }
                ]
            }
        }
    ]
}
```

1. From the Filters & transformers window, in the Get field, type **IP.AutoFocus.Resolutions.LastResolve**.



2. In the Filter section, click Add filter.

Cortex AgentIX automatically calculates that the context root to filter is **IP.AutoFocus.Resolutions**.

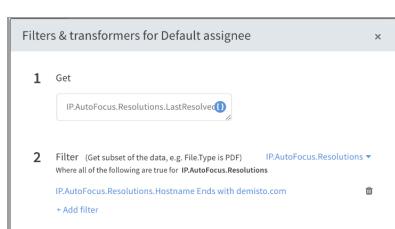


3. In the left-hand side, add **Hostname** to the filter.

4. Select Equals (String) â€“ Ends with

5. In the right-hand side add **demisto.com**.

6. Click the checkbox to save.



7. Click Test.



Create custom filters and transformers

If you require a filter or transformer that is not provided out-of-the-box, you can create your own by creating a script and then adding to the operators window.

1. Select Investigation & Response → Automation → Scripts → New Script.
2. Type a meaningful name for the script, and click Save.
3. To create a filter operator script, do the following:

- a. In the Tags field, add the `filter` tag.

If you want a custom transformer that operates on an entire array rather than on each individual item, you need to add the `entirelist` tag.

- b. In the Arguments section, add the following arguments:

Argument	Description
left	Mark as mandatory. This argument defines the left-side value of the transformer operation. In this example, this is the value being checked if it falls within the range specified in the right-side value.
right	Mark as mandatory. This argument defines the right-side value of the transformer operation. In this example, this is the range to check if the left-side value is in.

- c. Add the script syntax and save.

4. To create a transformer operator script do the following:

- a. In the Tags field, add the `transformer` tag.

- b. In the Arguments section, add the following arguments:

Argument	Description
value	Mark as mandatory. The value to transform. In this example, this is the UNIX epoch timestamp to convert to ISO format.

- c. Add the script syntax and save.

5. Go to the filters and transformers window and select the operator.

3.8.4.6.2.1 | Filter considerations, categories, and built-in filters

## Abstract

Filters in playbook tasks are defined built-in according to categories.

You can use built-in filters to define your filter, they are grouped by category. Before defining a filter, consider the following.



## Filter considerations

- Filters try to cast the transformed value and arguments to the appropriate type. The task fails if casting fails. For example, `a == Equals [some : object ]` => Error
- If the filter's left-side value expects a single item but receives a list, the filter passes if at least one item meets the requirements. For example, `[a, b, c] == b` => true.
- If the filter's left-side value expects a list but receives a single item, it converts it to a list with a single item. For example, `a == Contains a` => True.
- Some custom filters are implemented as scripts with the `filter` tag. You can find examples in the playbook automation task description.
- Filters in conditional tasks do not iterate the items of the root. Instead, they fetch the left-side value and the right-side value and compare them.

## Filter categories and built-in filters

When adding a filter, clicking the default Equals (String) field opens a search window showing the available built-in filters. They are defined by category as follows:

### General

General filters such as Contains, Doesn't Contain, In, and Is empty.

Filter	Description
Contains	Tests whether the value on the left is contained in the value on the right. Can be used for any kind of object (not limited to a string).
Doesn't Contain	Tests whether the value on the left is NOT contained in the value on the right. Can be used for any kind of object (not limited to a string).
Has length of	Tests whether a list specified on the left has the number of items specified on the right.
In	Tests whether the value on the left is contained in the object on the right.
Is defined	Tests whether a key on the left exists in context. <b>NOTE:</b> Is defined considers false and empty strings and lists to be defined values. If you don't want those to be included as defined, use Is not empty.
Is empty	Tests whether the value of a key is empty.
Is not empty	Tests whether the value of a key is NOT empty.
Not defined	Tests whether a key on the left does NOT exist in context. <b>NOTE:</b> Not defined considers false and empty strings and lists to be defined values. If you don't want those to be included as defined, use Is empty.
Not in	Tests whether the value on the left is NOT contained in the object on the right.

### String

Determines the relationship between the left-side string value and the right-side string value, such as starts with, includes, and in the list. The string filter returns partial matches as True.



Filter	Description
Doesn't end with	Tests whether the string on the left is NOT the end of the string on the right.
Doesn't equal	Tests whether the strings are NOT the same.
Doesn't include	Tests whether the string on the right is NOT a substring of the string on the left.
Doesn't start with	Tests whether the string on the right is NOT the beginning of the string on the left.
Ends with	Tests whether the string on the left is the end of the string on the right.
Equals	Tests whether the strings are the same.
Has length	Tests whether the two strings have the same length.
In list	Tests whether the string on the left is in the list on the right.
Includes	Tests whether the string on the right is a substring of the string on the left.
Matches - regex	Tests whether the string on the left matches the regex on the right. Uses Go-style regex.
Not in list	Tests whether the string on the left is NOT a substring of the string on the right.
Starts with	Tests whether the string on the right is the beginning of the string on the left.
StringContainsArray	Tests whether a substring or an array of substrings on the left is within a string array on the right. Supports single strings as well. For example, for substrings ['a', 'b', 'c'] in string 'a' the script returns true.

#### Number

Determines the relationship between the left-side number value and the right-side number value, such as Equals, Greater than, and Less than.

Filter	Description
Doesn't equal	Tests whether the number on the left does NOT equal the number on the right.
Equals	Tests whether the number on the left equals the number on the right.
Greater or equal	Tests whether the number on the left is greater than or equal to the number on the right.
Greater than	Tests whether the number on the left is greater than the number on the right.



<b>Filter</b>	<b>Description</b>
InRange	Tests whether the number on the left is within a range specified on the right. For example, if the left value is 4, and the range on the right is 1,8, the condition is true.
Less or equal	Tests whether the number on the left is less than or equal to the number on the right.
Less than	Tests whether the number on the left is less than the number on the right.

#### Date

Determines whether the left-side time value is earlier than, later than, or the same time as the right-side time value.

<b>Filter</b>	<b>Description</b>
After	Tests whether the date on the left is after the date on the right.
AfterRelativeDate	Tests whether the date on the left occurred after the provided relative time (such as '6 months ago') on the right. Returns True or False.
Before	Tests whether the date on the left is before the date on the right.
Same as	Tests whether the two dates are the same.

Supported time and date formats

<b>Format</b>	<b>Example</b>
ANSIC	Tues Jan _2 15:04:05 2019
UnixDate	Tues Jan _2 15:04:05 MST 2019
RubyDate	Tues Jan 02 15:04:05 -0700 2019
RFC822	02 Jan 19 15:04 MST
RFC822Z	02 Jan 19 15:04 -0700 // RFC822 with numeric zone
RFC850	Tuesday, 02-Jan-19 15:04:05 MST
RFC1123	Tues, 02 Jan 2019 15:04:05 MST
RFC1123Z	Tues, 02 Jan 2019 15:04:05 -0700 // RFC1123 with numeric zone
RFC3339	2019-01-02T15:04:05Z07:00



<b>Format</b>	<b>Example</b>
RFC3339Nano	2019-01-02T15:04:05.999999999Z07:00
Kitchen	3.04PM
Stamp	Jan _2 15:04:05
StampMilli	Jan _2 15:04:05.000
StampMicro	Jan _2 15:04:05.000000
StampNano	Jan _2 15:04:05.000000000

#### Boolean

Determines whether a field is true or false, or the string representation is true or false.

<b>Filter</b>	<b>Description</b>
Is false	Tests whether the value on the left evaluates to false.
Is true	Tests whether the value on the left evaluates to true.

#### Other

Miscellaneous filters, including CheckIfSubdomain and IsInCidrRanges.

<b>Filter</b>	<b>Description</b>
CheckIfSubdomain	Tests whether the value on the left is a subdomain of the value on the right.
CIDRBiggerThanPrefix	Tests whether the CIDR prefix on the left is bigger than the defined maximum prefix on the right.
GreaterCidrNumAddresses	Tests whether the number of available addresses in IPv4 or IPv6 CIDR on the right is greater than the input given on the left.
IsInCidrRanges	Tests whether the IPv4 address on the left is contained in at least one of the comma-delimited CIDR ranges on the right. Multiple IPv4 addresses can be passed in a comma-delimited list and each address is tested.
IsNotInCidrRanges	Tests whether the IPv4 address on the left is NOT contained in at least one of the comma-delimited CIDR ranges on the right. Multiple IPv4 addresses can be passed in a comma-delimited list and each address is tested.
IsRFC1918Address	Tests whether an IPv4 address on the left is in the private RFC-1918 address space (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16) on the right.
LowerCidrNumAddresses	Tests whether the number of available addresses in IPv4 or IPv6 CIDR on the right is less than the input given on the left.



## Abstract

Use transformers in playbook tasks according to the following considerations.

You can use built-in transformers to define your transformer, they are grouped by category. Before defining a transformer, consider the following.

### Transformer considerations

- Transformers try to cast the transformed value (and arguments) to the necessary type. Tasks will fail if casting has failed, for example `some : object` } To upper case => `Error`.
- Some transformers are applied on each item of the result. For example, `a, b, c` To upper case => `A, B, C`.
- Some transformers operate on the entire list. For example, `a, b, c` count => 3.
- Some custom transformers are implemented as scripts with the `transformer` tag. You can find examples in the playbook automation task description.

### Transformer categories and built-in transformers

When adding a transformer, clicking the default To upper case (String) field opens a search window showing the available built-in transformers. They are defined by category as follows.



Transformer Category	Description	Built-In Transformers												
General	Generic transformers	<p>General built-in transformers</p> <table border="1"> <thead> <tr> <th>Name</th><th>Description</th><th>Example</th></tr> </thead> <tbody> <tr> <td>Unique</td><td>Returns a deduped version of a list.</td><td><code>a, b, a, c, d, a, b =&gt; a, b, c, d</code></td></tr> <tr> <td>Slice</td><td> <p>Returns part of a specified list in a range of <code>from</code> index (included) through <code>to</code> index (not included)</p> <p><code>from</code>: Zero based index at which to begin extraction (default: 0).</p> <p><code>to</code>: Zero based index before which to end extraction (default: list length).</p> </td><td><code>a, b, c, d from: 1, to: 3 =&gt; b, c</code></td></tr> <tr> <td>Slice by item</td><td> <p>Returns part of a list specified in a range of <code>from</code> item (included) through <code>to</code> item (not included).</p> <p><code>from</code>: Item from which to begin the extraction. If not specified, extracts from the beginning of the list.</p> <p><code>to</code>: Item before which to end the extraction. If not specified, extracts from the end of the list.</p> </td><td><code>a, b, c, d from: b, to: d =&gt; b, c</code></td></tr> </tbody> </table>	Name	Description	Example	Unique	Returns a deduped version of a list.	<code>a, b, a, c, d, a, b =&gt; a, b, c, d</code>	Slice	<p>Returns part of a specified list in a range of <code>from</code> index (included) through <code>to</code> index (not included)</p> <p><code>from</code>: Zero based index at which to begin extraction (default: 0).</p> <p><code>to</code>: Zero based index before which to end extraction (default: list length).</p>	<code>a, b, c, d from: 1, to: 3 =&gt; b, c</code>	Slice by item	<p>Returns part of a list specified in a range of <code>from</code> item (included) through <code>to</code> item (not included).</p> <p><code>from</code>: Item from which to begin the extraction. If not specified, extracts from the beginning of the list.</p> <p><code>to</code>: Item before which to end the extraction. If not specified, extracts from the end of the list.</p>	<code>a, b, c, d from: b, to: d =&gt; b, c</code>
Name	Description	Example												
Unique	Returns a deduped version of a list.	<code>a, b, a, c, d, a, b =&gt; a, b, c, d</code>												
Slice	<p>Returns part of a specified list in a range of <code>from</code> index (included) through <code>to</code> index (not included)</p> <p><code>from</code>: Zero based index at which to begin extraction (default: 0).</p> <p><code>to</code>: Zero based index before which to end extraction (default: list length).</p>	<code>a, b, c, d from: 1, to: 3 =&gt; b, c</code>												
Slice by item	<p>Returns part of a list specified in a range of <code>from</code> item (included) through <code>to</code> item (not included).</p> <p><code>from</code>: Item from which to begin the extraction. If not specified, extracts from the beginning of the list.</p> <p><code>to</code>: Item before which to end the extraction. If not specified, extracts from the end of the list.</p>	<code>a, b, c, d from: b, to: d =&gt; b, c</code>												



Transformer Category	Description	Built-In Transformers		
		Name	Description	Example
		Sort	<p>Sorts an entire list. Supports strings and numbers.</p> <p>descending: <code>true</code> to sort in descending order, default is false.</p>	<code>b, c, a =&gt; a, b, c</code> <code>2.1, 1.2, 3.4 descending: true</code> <code>=&gt; 3.4, 2.1, 1.2</code>
		Get index	<p>Get item at the given index.</p> <p><code>index</code>: Index of the item to get.</p>	<code>b, c, a index: 0 =&gt; b</code> <code>b, c, a index -1 =&gt; nil</code>
		Splice	<p>Adds or removes items to/from an array.</p> <p><code>index</code>: (required) Zero-based index at which to begin add/remove items.</p> <p><code>deleteCount</code>: Number of elements to remove from <code>index</code>, default is 0.</p> <p><code>item</code>: Item to add to the array after <code>index</code> position.</p>	<code>a, b, c, d, index: 1 deleteCount: 2 =&gt; a, d</code> <code>a, b, c, d, index: 2 item: w</code> <code>=&gt; a, b, c, w, d</code>
		Index of	<p>Returns the first index of the element in the array, or <code>-1</code> if not found.</p> <p><code>item</code>: Item to locate in the array.</p> <p><code>fromLast</code>: <code>true</code> to get the index from last. (default is false).</p>	<code>a, b, a, c, d, a, b, item: b =&gt; 1</code> <code>a, b, a, c, d, a, b, item: a fromLast: true =&gt; 5</code> <code>a, b, a, c, d, a, b, item: w =&gt; -1</code>



Transformer Category	Description	Built-In Transformers		
		Name	Description	Example
		Get field	Extracts a given field from the given object.  <b>field:</b> (required) The field to extract from the result	{ name : john , color : white } field: color white
		Stringify	Converts the given item to a string.	{ name : john , color : white } => { name : john , color : white }
		Count	Returns the number of elements.	b, c, a => 3  null => 0  a => 1
		Join	Concatenates all elements.  <b>separator:</b> Specifies a string to separate each pair of adjacent elements of the array, default is an empty string.	b, c, a separator: , => b,c,a  b, c, a => bca



Transformer Category	Description	Built-In Transformers																					
String	<p>String transformers</p> <p><b>NOTE:</b> To make regex case non-sensitive, use the <code>(?i)</code> prefix (for example <code>(?<i>i</i>)yourRegexText</code>).</p>	<p>String built-in transformers</p> <table border="1"> <thead> <tr> <th>Name</th><th>Description</th><th>Example</th></tr> </thead> <tbody> <tr> <td>replace match</td><td> <p>Returns a string with some or all matches of a regex pattern, and replaces with a specified string.</p> <p>regex: A regex pattern to be replaced by the replaceWith argument.</p> <p>replaceWith: The string that replaces the string specified in the toReplace argument, default is an empty string. Detailed RegEx syntax can be found at <a href="https://github.com/google/re2/wiki/Syntax">https://github.com/google/re2/wiki/Syntax</a>.</p> </td><td> <pre>pluto,is,not,a,planet regex:    , replaceWith: ;  =&gt; pluto;is;not;a;planet</pre> <p>pluto is not a planet</p> <p>regex .*to replaceWith vega =&gt; vega is not a planet</p> </td></tr> <tr> <td>Substring</td><td> <p>Returns a subset of a string between one index and another, or through the end of the string.</p> <p>from (required): An integer between 0 and the length of the string, specifying the offset into the string of the first character to include in the returned substring.</p> <p>to (optional): An integer between 0 and the length of the string, which specifies the offset into the string of the first character not to include in the returned substring.</p> </td><td> <pre>pluto is not a planet from: 4 to: 10 =&gt; o is n</pre> </td></tr> <tr> <td>Split</td><td> <p>Splits a string into an array of strings, using a specified delimiter string to determine where to make each split.</p> <p>delimiter: Specifies the string which denotes the points at which each split should occur, default delimiter is ,.</p> </td><td> <pre>hello world,bye bye world =&gt; hello world, bye bye world</pre> <p>hello world delimiter</p> <p>=&gt; hello, world</p> </td></tr> <tr> <td>Split &amp; trim</td><td> <p>Splits a string into an array of strings and removes whitespace from both ends of the string, using a specified delimiter string to determine where to make each split.</p> <p>Arguments</p> <p>delimiter: Specifies the string which denotes the points at which each split should occur (default delimiter is , ).</p> </td><td> <pre>hello &amp; world delimiter: &amp; =&gt; hello, world</pre> </td></tr> <tr> <td>From string</td><td> <p>Returns a subset of a string from the first from string occurrence.</p> <p>from (required): String to substring from.</p> </td><td> <pre>pluto is not a planet from: pluto is =&gt; not a planet</pre> </td></tr> <tr> <td>To string</td><td> <p>Returns a subset of a string until the first to string occurrence.</p> <p>to (required): String to substring until.</p> </td><td> <pre>pluto is not a planet to: a planet =&gt; pluto is not</pre> </td></tr> </tbody> </table>	Name	Description	Example	replace match	<p>Returns a string with some or all matches of a regex pattern, and replaces with a specified string.</p> <p>regex: A regex pattern to be replaced by the replaceWith argument.</p> <p>replaceWith: The string that replaces the string specified in the toReplace argument, default is an empty string. Detailed RegEx syntax can be found at <a href="https://github.com/google/re2/wiki/Syntax">https://github.com/google/re2/wiki/Syntax</a>.</p>	<pre>pluto,is,not,a,planet regex:    , replaceWith: ;  =&gt; pluto;is;not;a;planet</pre> <p>pluto is not a planet</p> <p>regex .*to replaceWith vega =&gt; vega is not a planet</p>	Substring	<p>Returns a subset of a string between one index and another, or through the end of the string.</p> <p>from (required): An integer between 0 and the length of the string, specifying the offset into the string of the first character to include in the returned substring.</p> <p>to (optional): An integer between 0 and the length of the string, which specifies the offset into the string of the first character not to include in the returned substring.</p>	<pre>pluto is not a planet from: 4 to: 10 =&gt; o is n</pre>	Split	<p>Splits a string into an array of strings, using a specified delimiter string to determine where to make each split.</p> <p>delimiter: Specifies the string which denotes the points at which each split should occur, default delimiter is ,.</p>	<pre>hello world,bye bye world =&gt; hello world, bye bye world</pre> <p>hello world delimiter</p> <p>=&gt; hello, world</p>	Split & trim	<p>Splits a string into an array of strings and removes whitespace from both ends of the string, using a specified delimiter string to determine where to make each split.</p> <p>Arguments</p> <p>delimiter: Specifies the string which denotes the points at which each split should occur (default delimiter is , ).</p>	<pre>hello &amp; world delimiter: &amp; =&gt; hello, world</pre>	From string	<p>Returns a subset of a string from the first from string occurrence.</p> <p>from (required): String to substring from.</p>	<pre>pluto is not a planet from: pluto is =&gt; not a planet</pre>	To string	<p>Returns a subset of a string until the first to string occurrence.</p> <p>to (required): String to substring until.</p>	<pre>pluto is not a planet to: a planet =&gt; pluto is not</pre>
Name	Description	Example																					
replace match	<p>Returns a string with some or all matches of a regex pattern, and replaces with a specified string.</p> <p>regex: A regex pattern to be replaced by the replaceWith argument.</p> <p>replaceWith: The string that replaces the string specified in the toReplace argument, default is an empty string. Detailed RegEx syntax can be found at <a href="https://github.com/google/re2/wiki/Syntax">https://github.com/google/re2/wiki/Syntax</a>.</p>	<pre>pluto,is,not,a,planet regex:    , replaceWith: ;  =&gt; pluto;is;not;a;planet</pre> <p>pluto is not a planet</p> <p>regex .*to replaceWith vega =&gt; vega is not a planet</p>																					
Substring	<p>Returns a subset of a string between one index and another, or through the end of the string.</p> <p>from (required): An integer between 0 and the length of the string, specifying the offset into the string of the first character to include in the returned substring.</p> <p>to (optional): An integer between 0 and the length of the string, which specifies the offset into the string of the first character not to include in the returned substring.</p>	<pre>pluto is not a planet from: 4 to: 10 =&gt; o is n</pre>																					
Split	<p>Splits a string into an array of strings, using a specified delimiter string to determine where to make each split.</p> <p>delimiter: Specifies the string which denotes the points at which each split should occur, default delimiter is ,.</p>	<pre>hello world,bye bye world =&gt; hello world, bye bye world</pre> <p>hello world delimiter</p> <p>=&gt; hello, world</p>																					
Split & trim	<p>Splits a string into an array of strings and removes whitespace from both ends of the string, using a specified delimiter string to determine where to make each split.</p> <p>Arguments</p> <p>delimiter: Specifies the string which denotes the points at which each split should occur (default delimiter is , ).</p>	<pre>hello &amp; world delimiter: &amp; =&gt; hello, world</pre>																					
From string	<p>Returns a subset of a string from the first from string occurrence.</p> <p>from (required): String to substring from.</p>	<pre>pluto is not a planet from: pluto is =&gt; not a planet</pre>																					
To string	<p>Returns a subset of a string until the first to string occurrence.</p> <p>to (required): String to substring until.</p>	<pre>pluto is not a planet to: a planet =&gt; pluto is not</pre>																					



Transformer Category	Description	Built-In Transformers		
		Name	Description	Example
		concat	Returns a string concatenated with given prefix and suffix.  prefix: A prefix to concat to the start of the argument.  suffix: A suffix to concat to the end of the argument.	night prefix good => good night night suffix shift=> night shift
Number	Number transformers	Number built-in transformers		
		Name	Description	Example
		Floor	Returns the highest integer less than or equal to the number.	1.2=> 1
		Ceil	Returns the lowest integer greater than or equal to the number.	1.2 =>2
		Round	Returns the nearest integer, rounding half way from zero.	7.68 => 8 2.43 => 2 2.5 => 3
		Absolute	Returns the absolute value of the given number.	-2 => 2
		Decimal precision	Truncates the number of digits after the decimal point, according to the by argument.  by: Number of digits to keep after the decimal point, default is 0.	8.6666 by: 2 => 8.66
		Modulus (remainder)	The modular operator (%) returns the division remainder.  by (required): Modulo by, default:0	20 by: 3=> 2
		To percent	Converts a number to a percent.  withsign: Specify true to include %. Default is false	0.22 => 20 0.22 withsign: true =>20%
		Quadratic equation	Returns the result of the Quadratic Formula.b (required): The b number of: ax <sup>2</sup> + bx + c = 0, default is 0.c (required): The c number of: ax <sup>2</sup> + bx + c = 0, default is 0.	1 b: 3 c: 2=> -1.00, -2.00 3 b: 2 c: 4=> (-0.333 +1.106i), (-0.333 -1.106i)



Transformer Category	Description	Built-In Transformers																
Date	Date transformers	<p>Date built-in transformers</p> <table border="1"> <thead> <tr> <th>Name</th><th>Description</th><th>Example</th></tr> </thead> <tbody> <tr> <td>Date to string</td><td> <p>Converts any date to a specified string format. The date input must be in ISO format. For example, <code>2021-10-06T13:44:07</code>. The default output format is RFC822.</p> <p><b>format:</b> The desired string output format. For example, if you want to convert to RFC822 format, enter <code>02 Jan 06 15:04 MST</code>.</p> <p>The following are available output format options:</p> <ul style="list-style-type: none"> <li>• Layout = <code>01/02 03:04:05PM '06 -0700</code> // The reference time, in numerical order</li> <li>• RFC3339Nano = <code>2006-01-02T15:04:05.99999999Z07:00</code></li> <li>• Kitchen = <code>3:04PM</code> // Handy time stamps</li> <li>• Stamp = <code>Jan _2 15:04:05</code></li> <li>• StampMilli = <code>Jan _2 15:04:05.000</code></li> <li>• StampMicro = <code>Jan _2 15:04:05.000000</code></li> <li>• StampNano = <code>Jan _2 15:04:05.000000000</code></li> </ul> <p>This transformer is in GO language.</p> </td><td><code>2021-10-06T13:44:07 =&gt; 06 Oct 21 13:44 EDT</code></td></tr> <tr> <td>Date to Unix</td><td>Converts any date to Unix format.</td><td><code>Mon, 02 Jan 2006 15:04:05 MST =&gt; 1136214245</code></td></tr> </tbody> </table>	Name	Description	Example	Date to string	<p>Converts any date to a specified string format. The date input must be in ISO format. For example, <code>2021-10-06T13:44:07</code>. The default output format is RFC822.</p> <p><b>format:</b> The desired string output format. For example, if you want to convert to RFC822 format, enter <code>02 Jan 06 15:04 MST</code>.</p> <p>The following are available output format options:</p> <ul style="list-style-type: none"> <li>• Layout = <code>01/02 03:04:05PM '06 -0700</code> // The reference time, in numerical order</li> <li>• RFC3339Nano = <code>2006-01-02T15:04:05.99999999Z07:00</code></li> <li>• Kitchen = <code>3:04PM</code> // Handy time stamps</li> <li>• Stamp = <code>Jan _2 15:04:05</code></li> <li>• StampMilli = <code>Jan _2 15:04:05.000</code></li> <li>• StampMicro = <code>Jan _2 15:04:05.000000</code></li> <li>• StampNano = <code>Jan _2 15:04:05.000000000</code></li> </ul> <p>This transformer is in GO language.</p>	<code>2021-10-06T13:44:07 =&gt; 06 Oct 21 13:44 EDT</code>	Date to Unix	Converts any date to Unix format.	<code>Mon, 02 Jan 2006 15:04:05 MST =&gt; 1136214245</code>							
Name	Description	Example																
Date to string	<p>Converts any date to a specified string format. The date input must be in ISO format. For example, <code>2021-10-06T13:44:07</code>. The default output format is RFC822.</p> <p><b>format:</b> The desired string output format. For example, if you want to convert to RFC822 format, enter <code>02 Jan 06 15:04 MST</code>.</p> <p>The following are available output format options:</p> <ul style="list-style-type: none"> <li>• Layout = <code>01/02 03:04:05PM '06 -0700</code> // The reference time, in numerical order</li> <li>• RFC3339Nano = <code>2006-01-02T15:04:05.99999999Z07:00</code></li> <li>• Kitchen = <code>3:04PM</code> // Handy time stamps</li> <li>• Stamp = <code>Jan _2 15:04:05</code></li> <li>• StampMilli = <code>Jan _2 15:04:05.000</code></li> <li>• StampMicro = <code>Jan _2 15:04:05.000000</code></li> <li>• StampNano = <code>Jan _2 15:04:05.000000000</code></li> </ul> <p>This transformer is in GO language.</p>	<code>2021-10-06T13:44:07 =&gt; 06 Oct 21 13:44 EDT</code>																
Date to Unix	Converts any date to Unix format.	<code>Mon, 02 Jan 2006 15:04:05 MST =&gt; 1136214245</code>																
		Supported time and date formats																
		<table border="1"> <thead> <tr> <th>Format</th><th>Example</th></tr> </thead> <tbody> <tr> <td>ANSIC</td><td>Tues Jan _2 15:04:05 2019</td></tr> <tr> <td>UnixDate</td><td>Tues Jan _2 15:04:05 MST 2019</td></tr> <tr> <td>RubyDate</td><td>Tues Jan 02 15:04:05 -0700 2019</td></tr> <tr> <td>RFC822</td><td>02 Jan 19 15:04 MST</td></tr> <tr> <td>RFC822Z</td><td>02 Jan 19 15:04 -0700 // RFC822 with numeric zone</td></tr> <tr> <td>RFC850</td><td>Tuesday, 02-Jan-19 15:04:05 MST</td></tr> <tr> <td>RFC1123</td><td>Tues, 02 Jan 2019 15:04:05 MST</td></tr> </tbody> </table>	Format	Example	ANSIC	Tues Jan _2 15:04:05 2019	UnixDate	Tues Jan _2 15:04:05 MST 2019	RubyDate	Tues Jan 02 15:04:05 -0700 2019	RFC822	02 Jan 19 15:04 MST	RFC822Z	02 Jan 19 15:04 -0700 // RFC822 with numeric zone	RFC850	Tuesday, 02-Jan-19 15:04:05 MST	RFC1123	Tues, 02 Jan 2019 15:04:05 MST
Format	Example																	
ANSIC	Tues Jan _2 15:04:05 2019																	
UnixDate	Tues Jan _2 15:04:05 MST 2019																	
RubyDate	Tues Jan 02 15:04:05 -0700 2019																	
RFC822	02 Jan 19 15:04 MST																	
RFC822Z	02 Jan 19 15:04 -0700 // RFC822 with numeric zone																	
RFC850	Tuesday, 02-Jan-19 15:04:05 MST																	
RFC1123	Tues, 02 Jan 2019 15:04:05 MST																	



Transformer Category	Description	Built-In Transformers	
		Format	Example
		RFC1123Z	Tues, 02 Jan 2019 15:04:05 -0700 // RFC1123 with numeric zone
		RFC3339	2019-01-02T15:04:05Z07:00
		RFC3339Nano	2019-01-02T15:04:05.999999999Z07:00
		Kitchen	3.04PM
		Stamp	Jan _2 15:04:05
		StampMilli	Jan _2 15:04:05.000
		StampMicro	Jan _2 15:04:05.000000
		StampNano	Jan _2 15:04:05.000000000

#### 3.8.4.6.3 | Extract indicators

##### Abstract

Extract indicators from Cortex AgentiX issue fields and enrich them with commands and scripts.

In Cortex AgentiX, the indicator extraction feature extracts indicators from issue fields and enriches them using commands and scripts.

For more information about indicator extraction, see Extract and enrich an indicator.

How to set up indicator extraction in a playbook task

1. Select the playbook where you want to add indicator extraction, and click Edit.
2. In the playbook, click a task to open the Task Details pane.
3. Click the Advanced tab.
4. For Indicator Extraction mode, select the mode you want to use (default is none).
5. Click OK.

Example 31.

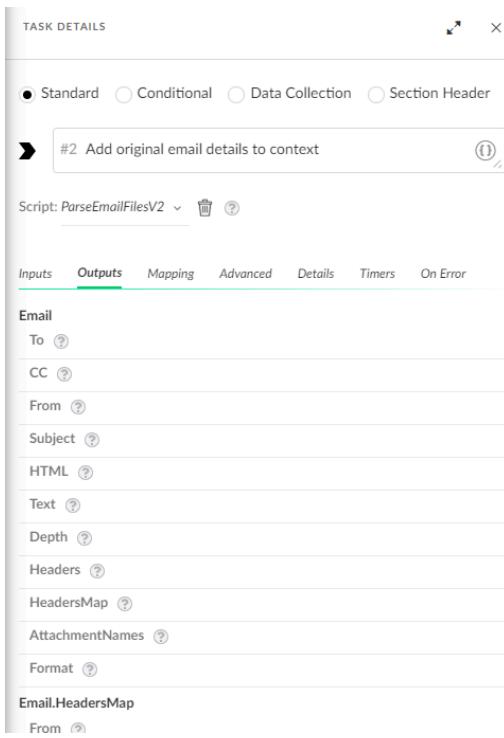
The following scenario shows how indicator extraction is used in the Process Email - Generic v2 playbook to extract and enrich a very specific group of indicators.

This playbook parses the headers in the original email used in a phishing attack. It is important to parse the original email used in the phishing attack and not the email that was forwarded to ensure that you only extract the email headers from the malicious email and not the one your organization uses to report phishing attacks.

1. Navigate to the Playbooks page and search for the Process Email - Generic v2 playbook.
2. Click  and select either Duplicate (create a copy of the playbook to edit) or Edit Playbook (detach the playbook).
3. Open the Add original email details to context task, and for the Script drop down, change the script from Set to ParseEmailFilesV2.

Under the Outputs tab, you can see all of the different data that the task extracts.





4. Click the Advanced tab and set Indicator Extraction mode to **Inline**. This ensures all the outputs are processed before the playbook moves ahead to the next task.
5. Open the Display email information in layout - Email.Headers task. This task receives the data from the saved attachment tasks and sets the various data points to context.
6. Click the Advanced tab and set Indicator Extraction mode to **None**, because the indicators were already extracted earlier in the Extract email artifacts and attachments task and there is no need to extract them again.

#### Indicator extraction modes

Indicator extraction supports the following modes:

- **None**: Indicators are not extracted automatically. Use this option when you do not want to further evaluate the indicators.
- **Inline**: Indicators are extracted within the context that indicator extraction runs (synchronously). The findings are added to the context data. For example, if indicator extraction for a playbook task is inline, extraction occurs before the next playbook tasks run.

#### **NOTE:**

This configuration may delay playbook execution (issue creation).

While indicator creation is asynchronous, indicator extraction and enrichment are run synchronously. Data is placed into the issue context and is available via the context for subsequent tasks.

- **Out of band**: Indicators are extracted in parallel (asynchronously) to other actions. The extracted data will be available within the issue, however, it is not available for immediate use in task inputs or outputs because the information is not available in real-time.

#### **NOTE:**

When using out of band, the extracted indicators do not appear in the context. If you want the extracted indicators to appear select inline.

- If system-wide indicator extraction is enabled, indicators are extracted according to the following rules:
  - Issue creation - inline
  - Issue field change - inline
  - Tasks - none, can be overridden on a per task basis
  - CLI - out of band, but can be overridden on a per-command basis

#### Troubleshoot indicator extraction

If indicators are not extracted, check whether the indicator mode is set to none. Even if you select the relevant issue fields and the indicators to extract, if the mode is set to none, indicators do not extract.



## Abstract

Extend context to retrieve specific information from integrations or commands and map to fields.

By design, integrations do not write all of the data returned from a command to the context. This prevents large context size and enables you to store only the most relevant information.

The Extend Context feature enables you to save additional data from the raw response of the command. For example, when a command runs to retrieve events from a SIEM, only some of the event fields are written to context, according to the integration design. With Extend Context, you can save additional fields specific to your use case.

Extend Context can also be used when the same command runs multiple times in the same playbook, but the outputs need to be saved to different context keys. For example, you can execute the `!ad-get-user` command twice, once to retrieve the user's information and again to retrieve the user's manager's information. By default, an integration command writes the data from the same command to the same context key. By using Extend Context, you can write the command's response to a custom context key of your choice.

You can extend context either in a playbook task or directly from the command line. Whichever method you use, first run your command with the `raw-response=true` flag. This helps you identify the information that you want to add to your extended data.

### Filter for specific keys from lists of dictionaries

You can use DT to get select keys of interest from a command that returns a list of dictionaries containing many keys. For example, the `findIndicators` automation returns a long list of indicator properties, but you may only be interested in saving the value and the indicator\_type to minimize the size of the context data. For more information about DT, see Cortex XSOAR Transform Language (DT).

#### Example 32.

- Run the command `!findIndicators size=2 query="type:IP" raw-response=true`.

You will see a list of two dictionaries containing 20+ items.

- Use the following value for extend-context to save only value and indicator\_type into a context key called FoundIndicators:

```
!findIndicators size=2 query="type:IP" extend-context="FoundIndicators=[{"value": val.value, "indicator_type": val.indicator_type}]`
```

- Use the following value for extend-context to save only the issue name, status, and id to a key called FoundIssues:

```
!SearchIssuesV2 id=<ANY_ISSUE_ID> extend-context="FoundIssues=Contents.data=[{"name": val.name, "status": val.status, "id": val.id}]` ignore-outputs=true
```

### Extend context in a playbook task

- Go to the **Advanced** tab of the relevant playbook task, such as a Data Collection task.
- In the Extend Context field, enter the name of the field in which you want the information to appear and the value you want to return. For example, using the `!ad-get-user` command, enter `name="john" attributes=displayname` to place the user's name in the `displayName` key.

The following image shows the result of the `!IPReputation ip=20.8.1.5 raw-response=true` command.



## #INCIDENT-7898 -

```
DBot
Feb 14th 2022 13:26:40
!IPReputation ip="20.8.1.5" raw-response="true"

Search in JSON entries

root: 3 items
  bucketInfo: 7 items
    dailyBucketStart: 2022-02-13 18:57:20
    dailyPoints: 100000
    dailyPointsRemaining: 82629
    minuteBucketStart: 2022-02-14 11:25:48
    minutePoints: 200
    minutePointsRemaining: 159
    waitInSeconds: 0
  indicator: 8 items
    firstSeenTsGlobal: null
    indicatorType: IPV4_ADDRESS
    indicatorValue: 20.8.1.5
    lastSeenTsGlobal: null
      latestPanVerdicts: 1 item
        PAN_DB: UNKNOWN
      seenByDataSourcelds: 0 items
      summaryGenerationTs: 1644838000403
      wildfireRelatedSampleVerdictCounts: 0 items
  tags: 0 items
```

To include more than one field, separate the fields with a double colon. For example: `attributes=displayName::manager=attributes.manager`

3. To output only the values for Extend context and ignore the standard output for the command, select the Ignore Outputs checkbox.

While this will improve performance, only the values that you request in the Extend Context field are returned. You cannot use Field Mapping as there is no output to which to map the fields.

Extend context using the CLI

1. Run your command with the extend-context flag `!<commandName><argumentName> <value>extend-context=contextKey=JsonOutputPath`.

For example, to add the user and manager fields to context use the ad-get-user command, as follows:

```
!ad-get-user=${user.manager.username} extend-context=manager=attributes.manager::attributes=displayName
```

2. To output only the values that you set as Extend context, run the command with the ignore-output flag=true. `!ad-get-`

```
user=${user.manager.username} extend-context=manager=attributes.manager::attributes=displayName ignore-
output=true
```

3.8.4.6.5 | Update issue fields with playbook tasks

Abstract

Use the setIssue script to set and update all system issue fields.

During the investigation you can set and update issue fields using the setIssue script in a playbook task.

You initially define issue fields after the planning stage, with mapping and classification for how the issues will be ingested from third-party integrations into Cortex AgentX.

### NOTE:

- The setIssue script includes all available fields; use the scroll bar to see all the fields.
- The name field has a limit of 600 characters. If there are more than 600 characters, you can shorten the name field to under 600 characters and then include the full information in a long text field such as the description field.
- There are many fields already available as part of the Common Type content pack. Before creating a new issue field, check if there is an existing field that matches your needs.

For more information, see Update issue fields



## Abstract

Generic Polling playbook enables you to periodically poll the status of a process on a remote host.

When working with third-party products (such as detonation, scan, search, and other third-party products) you may need to wait for a process to finish on the remote host before continuing. In these cases, the playbook should stop and wait for the process to complete on the third-party product, and continue when it is done. Integrations or automations may not be able to do this due to hardware limitations.

Generally, polling is used in the following scenarios:

- File detonation in a sandbox
- URL detonation
- Queries that take a long time to complete

To use polling, Cortex AgentX comes out-of-the-box with the GenericPolling playbook, which periodically polls the status of a process being executed on a remote host, and when the host returns that the process execution is done, the playbook finishes execution. For more information about using this playbook, see [Generic Polling](#).

The GenericPolling playbook is used as a sub-playbook to block the execution of the main playbook until the remote action is complete. There are a number of playbooks that use the GenericPolling playbook that come out-of-the-box or installed from a content pack, such as:

- Context Polling - Generic: Polls a context key to check if a specific value exists.
- Field Polling - Generic: Polls a field to check if a specific value exists.
- Scan Site - Nmap: Scans according to asset IP addresses or host names from Rapid7 Nmap, and waits for the scan to finish by polling the scan status in pre-defined intervals.

### NOTE:

You need to use the GenericPolling playbook as a sub-playbook in a main playbook, such as Detonate File - JoeSecurity.

The main playbook should follow this structure:

1. **Start Command:** The task contains a command that fetches the initial state of the process and saves it to context. This command starts the process that should be polled. For example:

Detonation: Submits a sample for analysis (detonated as part of the analysis), using the `joe-analysis-submit-sample` command.

Scan: Starts a scan for specified asset IP addresses and host names using the `nmap-start-assets-scan` command.

Search: Searches in QRadar using AQL using the `qradar-searches` command.

2. **Polling Command:** The task contains the GenericPolling sub-playbook that polls for an answer. For example:

- Detonation: After the file is submitted to Joe Security, the playbook polls for specific information for analysis, such as ID, status, comments, errors, SHA-256 hash details.
- Scan: After the scan runs in Nmap, using the playbook polls for scan information such as the scan type, the number of assets found, the scan ID, and other information.
- Search: The playbook runs the `qradar-get-search` to poll for the search ID and status.

3. **Results Task:** Returns the results of the operation. The task contains the results that were polled, which are added to context. For example, after polling JoeSecurity, the results are added to context.

For information about the GenericPolling playbook inputs such as `Ids`, `Interval`, and `dt`, see [Playbook inputs](#).

Example 33.

This generic polling example uses the Detonate File - JoeSecurity playbook from the Joe Security content pack.

The Detonate File - JoeSecurity playbook detonates one or more files using the Joe Security integration and returns relevant reports to the War Room and file reputations to the context data.

1. If you have not done so, go to Marketplace and download the Joe Security content pack.
2. Go to Playbooks and search for Detonate File - JoeSecurity.
3. Open the JoeSecurity Upload File task. This task uses the `joe-analysis-submit-sample` command, which starts a new analysis of a file in Joe Security. This is the **Start** command.
4. Open the GenericPolling task. This is the **Polling** command.

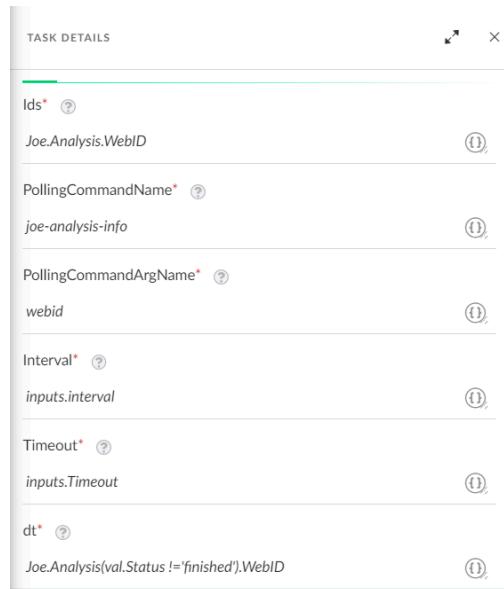


- **Ids**: Returns a list of `Joe.Analysis.ID`'s to poll.
- **PollingCommandName**: The `joe-analysis-info` command returns information for a specified analysis, such as status, MD5, SHA256, vendor.
- **PollingCommandArgName**: The `webid` argument name of the polling command.
- **dt**: The filter for polling. This is defined as `Joe.Analysis(val.Status!=='finished').ID`.

`Joe.Analysis`: The object to return.

`(val.Status != 'finished').ID` Gets the object that has a status other than 'finished', and then gets its ID field. The polling is done only when the result is `finished`. When finished, the `dt` filter returns an empty result, which triggers the playbook to stop running.

You can change the `Status` to: `starting`, `running`, or `finished`.



5. Open the JoeSecurity Get Info task. The `joe-analysis-info` command returns details of the IDs that have finished polling. This is the **Results** task.

6. Open the Set Context task. The context path to store the poll results is `Joe.Analysis`.

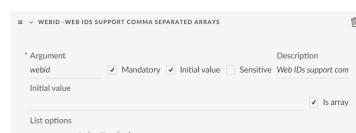
#### GenericPolling playbook limitations

The GenericPolling playbook has the following limitations.

- Global context is not supported.

Global context outputs enable receiving information from multiple integrated products when executing playbooks and commands.

- It does not run from the Playground.
- It uses the `ScheduleGenericPolling` script, which must support a list argument.



#### Troubleshoot playbook polling

The following are common generic polling issues and the recommended ways to deal with them.



- The playbook is stuck on **Waiting for polling to complete**.

As generic polling schedules tasks are outside the context of the playbook (not visible in the playbook run), errors may appear only in the Case War Room. Go to the War Room of the case and check for errors or warnings related to GenericPolling tasks.

- The GenericPolling task completes but the status has still not "finished".

If the timeout is reached, the playbook successfully finishes even if there are items that did not complete. Try increasing the timeout value for the GenericPolling task.

- The integration returns an ID not found error when running from the GenericPolling sub-playbook, but when running manually, it finishes successfully.

Some products cannot handle consecutive requests to query an action status right after the request to perform the action. After you initiate the action, try adding a Sleep task before calling the GenericPolling sub-playbook.

### 3.8.4.7 | Test your playbook

#### Abstract

Set breakpoints, conditional breakpoints, skip tasks, and input and output overrides in the playbook debugger.

The debugger provides a test environment where you can make changes to data and playbook logic and view the results in real-time to test and troubleshoot playbooks. You can see exactly what is written to the context at each step and which indicators are extracted.

To open a detached system playbook, a copy of a system playbook, or a custom playbook in the debugger, select the playbook and click Edit.

To open an attached playbook in the debugger, select the playbook and click View to access the debugger. While editing a playbook, sub-playbooks can be opened directly in the debugger by choosing Open sub-playbook in the task pane.

In some cases, you may have a playbook that includes two or more copies of the same sub-playbook. When you set breakpoints, override inputs or outputs, or skip tasks in sub-playbook A, the same changes apply to the identical sub-playbook B. In addition, if you set a breakpoint, override inputs or outputs, or skip tasks within a loop in a playbook, that setting will be applied every time the loop executes.

Running the debugger involves the following actions.

#### Choose test data

The debugger uses test data to execute the playbook, so you can see what your expected results would be. The following are options for test data.

#### **NOTE:**

The debugger does not support using **parentIncidentFields**.

- New Mock Issue:** By default, the debugger runs using an empty mock issue. An empty mock issue is useful to test simple functionality, such as a playbook that does simple tasks such as parsing inputs.
- Existing Issue:** You can select an existing issue. Using an existing issue in the debugger does not change the original issue. Click the Debugger Panel and in the Test data field, select an existing issue. The last fifty issues appear in the drop-down, as well as any issues you own or are a member of, or that you have participated in.

#### **NOTE:**

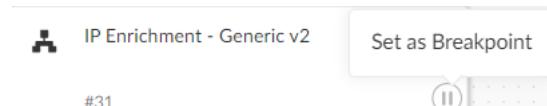
Using an existing issue in the debugger does not affect the original issue or change the original context data.

#### Set a breakpoint

At the breakpoint, you can override inputs and outputs to see how changes affect playbook execution. In addition, conditional breakpoints set conditions for the playbook to proceed. The playbook only pauses if your condition is met, letting you manipulate data to see how different scenarios impact how the playbook runs. For example, you can set a conditional breakpoint to pause the playbook when a phishing issue targets a member of a VIP asset list. If there are no VIPs in this issue, the execution does not pause. If there is a VIP in the issue, you can check that the member was properly identified by the playbook task.

Breakpoints do not apply to manual tasks, as a manual task will always pause the playbook run unless you skip the manual task. When the playbook reaches a breakpoint, no new tasks begin, but parallel tasks that have already begun continue. Breakpoints can be set in both the parent playbook and sub-playbooks.

- To set a breakpoint, go to a task and click on the breakpoint button. When a breakpoint is set, the breakpoint button changes to orange.



- After a breakpoint is reached, click the task to override inputs and outputs if needed.

- When you are finished with the task, run the debugger, and in the task, select an option for the playbook to continue.

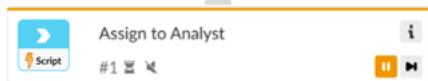


For an automated task, you have the options Run automation now or Complete Manually. If you choose Complete Manually, click on Mark Completed for the playbook to continue.

For a task that is a sub-playbook, click Run playbook now for the playbook to continue.

For a conditional task, choose which branch the playbook should follow and click Mark Completed for the playbook to continue. The default branch is else.

When the playbook reaches a breakpoint, the task has an orange line at the top to indicate the breakpoint.



Breakpoint alerts are also displayed at the top of the playbook, enabling you to navigate between multiple breakpoints that have been reached in the playbook or sub-playbooks.

#### Start and stop the debugger

The debugger runs the playbook with the permissions of the logged in user. If a user runs potentially harmful commands, they are logged to the audit trail with the user's username. When the user sets breakpoints, skips tasks, or overrides inputs or outputs, those changes only apply to the individual user's session and do not permanently change the playbook. Using an existing issue as test data does not affect the original issue or change the original context data. When tasks run, however, they execute the same as they would without the debugger. For example, if you run the debugger and a task adds an item to a list, that item will be in the real list, accessible across for all users with permission to view that list.

Breakpoints pause playbook execution before a specific task. When the playbook is paused, the Debugger Panel displays the current state of context data, indicators, and task information.

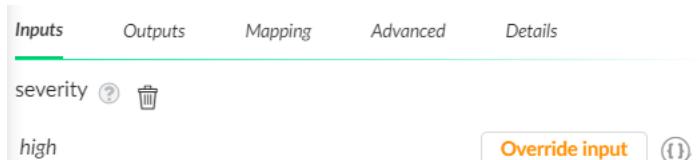
To start the debugger, click Run. When you click Stop, the debugger stops, and the context data is reset to the original issue data. In the case of a new mock issue, the context data is cleared and the context is empty. Any breakpoints, skips, or overrides you applied are still available.

#### Override inputs and outputs

The debugger enables you to temporarily override inputs and outputs for a playbook run and to view the results in real time. When you override an input or output in the debugger, the change is saved only in the debugger view and only for the user who made the change. If after testing you decide to keep the temporary changes you made and apply them permanently to the playbook for all users, you need to cancel the override and edit the task. Tasks can be edited directly in the debugger or outside of the debugger using the standard playbook editing options.

You can override task inputs or outputs before or during a playbook run to troubleshoot tasks that fail or to try different input and outputs as part of playbook development. If you override an input or output during a playbook run, the override is applied to the run if the playbook has not yet reached that task. If you edit (permanently change) inputs during a playbook run, the changes only take effect the next time you run the playbook. You cannot use filters or transformers for overrides.

1. To override an input or output, open the task and hover over any existing input or output. Click Override Input.



2. Enter a new input or output that will be used only in the debugger. For output overrides, you can enter a value, an array of values, or JSON. For input overrides, you can only enter plain text.

3. Click OK to save your changes.

The playbook task card displays a label indicating that the task input or output has been overridden.

#### Skip tasks

For testing purposes, you may want to skip a task that for example closes a port in a firewall, deletes an email, or sends a notification to a manager. Or you might skip a task where the integration has not yet been configured. By skipping a task and overriding the output, you can provide the data necessary to complete the playbook run. When you skip a conditional task, you can choose which branch runs after the skipped task, enabling you to test different outcomes for multiple branches.

You might need to skip tasks within a playbook:



- To check if a particular task is causing an issue.
- To avoid performing tasks not relevant for your troubleshooting.
- To skip tasks with potentially harmful results such as blocking a user or opening a port in a firewall.
- To skip tasks for integrations that are not yet configured.

#### How to skip a task

1. Click the  button for the task.

When a task is set to skip, the  button will be orange.



2. If the output is required for the playbook to proceed, click the task and override inputs and outputs.

[View context data, indicators, and task information](#)

Within the debugger panel, you can view the context data during the playbook run as well as the indicators as they are extracted by clicking any completed task in the playbook while the debugger is running.

You can see the results of that task in the debugger panel.

The screenshot shows the 'DEBUGGER PANEL' window. At the top, there are tabs: 'Task #1' (which is selected), 'Context', and 'Indicators'. Below the tabs, a section titled 'RESULTS (1)' is expanded. Inside this section, there is a card for 'DBot' from 'Feb 15th 2022 09:15:57'. The card contains the 'Task Result #1: print' information, which includes the command '!Print value="Hello Google " (Scripts)' and the output 'Hello Google'.

#### 3.8.4.7.1 | Troubleshoot playbook performance

##### Abstract

Obtain playbook metadata to troubleshoot performance issues.

You can analyze playbook metadata such as tasks input and output, the amount of storage each task input/output uses, and the type of task. This is useful when troubleshooting your custom playbook if your system has slowed down and is using high CPU usage, memory, or storage (disk space).

##### Get data from XQL datasets

You can leverage XQL for flexible and adjustable playbook and script tracking to provide performance and execution data for debugging. The following datasets are available for querying and dashboards:

- `playbook_tasks`: Data about task executions within playbooks.
- `playbook_runs`: Data about playbook runs and statuses.
- `scripts_and_commands_metrics`: Data about scripts and commands used in playbook tasks.

##### Get playbook metadata using the CLI

After an issue has been assigned to a playbook you can analyze it to see its tasks inputs/outputs storage. You can filter the data according to the KB used in each task input/output.

From the Cases & Issues  Cases page, in the Case War Room tab the following command in the CLI.

```
!getInvPlaybookMetaData issueId=<issue ID> minSize=<size of the data you want to return in KB. Default is 10>
```

To view the playbook metadata that is used in issue number 964, in the CLI type `!getInvPlaybookMetaData incidentid=< 964> minSize=< 0> !getInvPlaybookMetaData incidentid=< 964> minSize=< 0> .`



## Troubleshooting Playbooks dashboard

From the Troubleshooting Playbooks dashboard, you can view playbook and task errors, average playbook run time, and execution by status for manual and automated tasks. You can also pivot to the XQL view for more detailed data analysis.

### 3.8.4.8 | Manage playbook content

#### Abstract

Manage playbook content and avoid concurrent playbook editing.

##### Manage playbook content with a remote repository

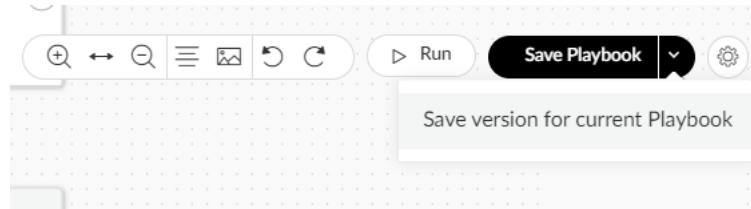
In Cortex AgentiX, you can develop and test your playbook content on development machines before using it in a production environment using the remote repository feature.

For more information about content management in Cortex AgentiX, see Cortex AgentiX development tenant.

##### Save versions of your playbook in Cortex AgentiX

You can save versions of a playbook as you are developing it. When you save a version of a playbook, add a meaningful comment so that you will be able to recognize the changes you made in that version at a later time. The version is saved with the name of the playbook, your commit message, an indication of what the change was (modify, insert), the date the playbook was saved, and the name of the author who last saved it. If necessary, you can access the playbook's version history and revert your playbook to a previous version.

1. In a playbook, after making changes, click the list next to Save Playbook and then click Save version for current Playbook.

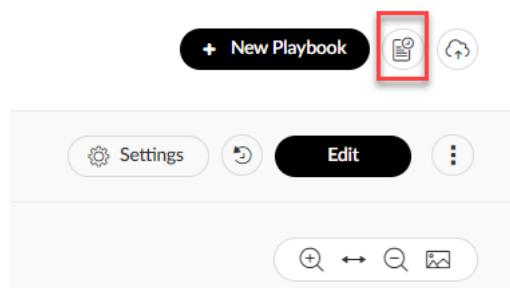


2. Enter a description of the change that was made to the current version.

3. Click Update Playbook.

4. To access a version of a playbook:
  - a. Click the icon next to New Playbook. The tooltip displays Version history for all Playbooks.

a. Click the icon next to New Playbook. The tooltip displays Version history for all Playbooks.



- b. Search for the required playbook. The description that was entered when the version was saved should help you locate the version you now require.

- c. Click Restore to restore the required version of the playbook.

##### Playbook editing conflict management

If multiple users simultaneously attempt to edit the same playbook, they could overwrite each other's changes in the playbook editor. To prevent users from accidentally losing their work, the playbook editor only allows the first user to edit and save. Subsequent users are automatically placed in View Mode Only, and a banner in the playbook editor clearly shows who is currently editing the playbook. The subsequent users can still view, debug, run, duplicate, and download the playbook.

#### NOTE:

Opening a playbook does not immediately lock it. A playbook is considered to be in edit mode only when a user makes a modification that causes the Save button to become available.

The playbook is automatically unlocked when the user currently editing the playbook saves the playbook, closes the editor, logs out, or when their session expires. In addition, users with playbook View/Edit and Unlock permission can click Unlock in the banner on the Playbooks page or directly in the playbook



editor to force-unlock the playbook.

#### **IMPORTANT:**

Manually unlocking a playbook may cause version conflicts. This turns off concurrent editing protection, and if the original user is still editing, their changes might be lost or overwritten.

The playbook Unlock permission is located under Settings → Configurations → Access Management → Roles. Edit a role and go to Investigation & Response → Automations → Playbooks. Ensure you have View/Edit permissions selected.

#### 3.8.4.9 | Best practices

##### Abstract

Best practices for building and working with playbooks.

The following guidelines are best practices for building playbooks as well as optimizing playbook design and performance. Whether you are just starting or are creating advanced workflows, we recommend reviewing these recommendations carefully so your playbooks have a clear logical flow and run correctly and efficiently.

Best practices for building your playbook

##### Use clear task names and descriptions

Describe tasks clearly. Tasks should be clear to someone not familiar with the playbook workflow. This applies to task names, task descriptions, and the playbook description. When naming tasks, the guideline should be that users can understand what the playbook does by reading the task names, without having to open individual tasks to view the details.

Clear	Unclear
Task name: Check if the IP is Private	Task name: IP Check

##### Define playbook inputs and outputs properly

- **Group related input fields.**

Grouping inputs organizes the input fields and provides clarity and context to understand which inputs are relevant to which playbook flow.

- **Use Pascal case for input names.**

Use the PascalCase convention for inputs, keeping in mind that inherently capitalized terms should be kept in upper case. For example, the `Entity ID` input should be named `EntityID` and `MITRE Technique` should be `MITRETechnique`.

- **Define outputs properly.**

When configuring playbook outputs, configure sub-keys as much as possible, do not limit configuration to only the root keys. For example, instead of outputting `File`, output `File.Name`, `File.Size`, etc. This helps when viewing the outputs of the playbook within another playbook.

##### Configure playbook task inputs correctly

- **Avoid using Cortex AgentX Transform Language (DT) in the Get input field definition.**

If you need to use DT for complex processing and you think a new filter or transformer would provide a better alternative to your DT solution, you can request the feature or contribute it. Consider using DT only if it can drastically simplify the playbook or improve performance.

##### Define playbook logic carefully

In each task, make sure appropriate logical operations are performed on input data. For example:



- Avoid race conditions.

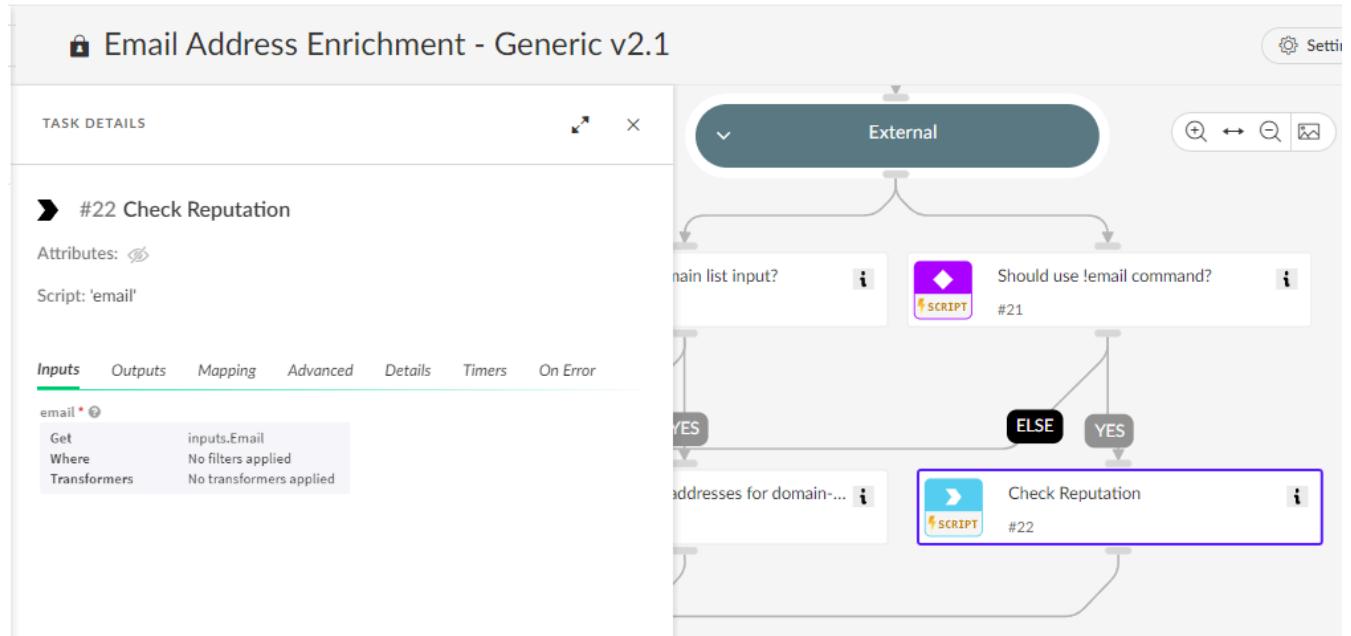
Be aware of potential race conditions. When you want to add multiple values to the same key, do not use multiple tasks that run `Set`, `SetAndHandleEmpty`, or any other script that sets data in context at the same time, because a race condition can cause your data to be overwritten by the same tasks. This is especially problematic when trying to append data. Instead, run the tasks one after the other or use scripts to append the data instead of setting a new value to the key.

- Determine where inputs are coming from.

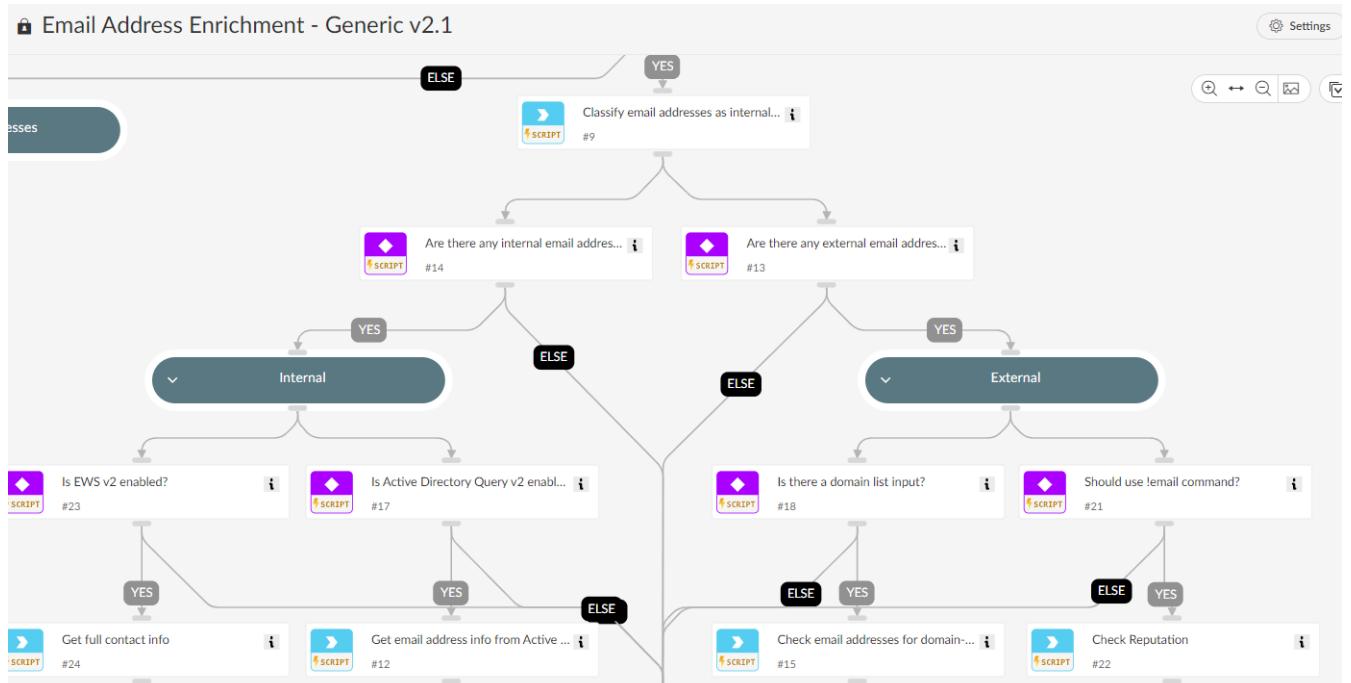
Verify whether the data you're getting is `As value` (simple value) or `From Previous Tasks` (from context).

- Filter your inputs correctly so the task runs efficiently.

Tasks take their inputs from the context, not directly from the previous tasks (even if it says from previous tasks). For an example of a task not receiving the right context, see this bug (since fixed) in a playbook:



The playbook begins by classifying the emails as internal or external. It then checks the reputation of external email addresses if any were found. That happens on the right side of the image. We expect that branch to run only if external addresses are found.



However, we did not apply a filter to the last task that gets the reputation on the right side:

This means that if both internal and external email addresses are found, we proceed with both branches (internal and external) of the playbook, and the task that gets the reputation runs without an applied filter, effectively taking all the emails we have in the inputs. The correct task input should have been:



email \*

Get	Account.Email.Address
Where	Account.Email.NetworkType Equals External
Transformers	Unique

- Select Ignore case for input names.

Use `ignore-case` option where possible, especially when checking Boolean playbook inputs such as `True` which users may end up configuring as `true` with a lowercase t:

The screenshot shows a search bar at the top with the placeholder "Search...". Below it is a table of comparison operators categorized by type: General, String, Number, and Boolean. A red box highlights the "Ignore case" checkbox at the bottom left of the table.

General	String	Number
Contains	Doesn't end with	Doesn't equal
Doesn't Contain	Doesn't equal	Equals
Has length of	Doesn't include	Greater or equal
In	Doesn't start with	Greater than
Is defined	Ends with	InRange
Is empty	Equals	Less or equal
Is not empty	Has length	Less than
Not defined	In list	

Ignore case

- When working with two lists, if you need multiple items from list A, which are also in list B, use the `in` filter instead of the `equals` or `contains` filters.

Correct Method	Incorrect Method						
<p>Get the IP addresses that <code>are</code> in the list of inputs.</p> <p>ip *</p> <table border="1"> <tr> <td>Get</td> <td>IP</td> </tr> <tr> <td>Where</td> <td>IP.InRange Equals no</td> </tr> <tr> <td>Transformers</td> <td>IP.Address In inputs.IP Get field (field: Address) Unique</td> </tr> </table>	Get	IP	Where	IP.InRange Equals no	Transformers	IP.Address In inputs.IP Get field (field: Address) Unique	<p>Get the IP addresses where the addresses <code>contain</code> the list. This is incorrect because they don't contain the list, they contain individual items from it.</p>
Get	IP						
Where	IP.InRange Equals no						
Transformers	IP.Address In inputs.IP Get field (field: Address) Unique						

- Differentiate between checking if a `specific element exists` versus checking if an element equals something. This is a common mistake that can lead to tests working in some situations, but not all.

Correct Method	Incorrect Method												
<p>Check if <code>any object</code> where the NetworkType is External <code>exists</code>.</p> <p>Condition for: yes</p> <table border="1"> <tr> <td>Get</td> <td>IP</td> </tr> <tr> <td>Where</td> <td>IP.NetworkType Equals External</td> </tr> <tr> <td>Transformers</td> <td>No transformers applied</td> </tr> </table> <p>Is not empty (General)</p>	Get	IP	Where	IP.NetworkType Equals External	Transformers	No transformers applied	<p>Check if the NetworkType <code>of the IP object is External</code>. This is incorrect because the IP object may contain multiple IPs, some internal and some external.</p> <table border="1"> <tr> <td>Get</td> <td>IP.NetworkType</td> </tr> <tr> <td>Where</td> <td>No filters applied</td> </tr> <tr> <td>Transformers</td> <td>No transformers applied</td> </tr> </table> <p>Equals (String)</p> <p>Get External</p> <p>+ And</p>	Get	IP.NetworkType	Where	No filters applied	Transformers	No transformers applied
Get	IP												
Where	IP.NetworkType Equals External												
Transformers	No transformers applied												
Get	IP.NetworkType												
Where	No filters applied												
Transformers	No transformers applied												

- Run one or more tasks based on the object types versus running either one task or the other based on the type of one object.



Correct Method	Incorrect Method
<p>Check the existence of both object types and run tasks for the types found.</p> <pre> graph TD     A[Are there Internal IPs? #1] -- YES --&gt; B[Do stuff #4]     C[Are there External IPs? #2] -- YES --&gt; D[Do stuff #5]     B --&gt; E[Continue #3]     D --&gt; E   </pre>	<p>Check if there is either an internal or an external IP, and take only one path even if both types exist.</p> <pre> graph TD     A[Filter IPs by range #7] --&gt; B[INTERNAL]     A --&gt; C[EXTERNAL]     B --&gt; D[Do stuff #8]     C --&gt; E[Do different stuff #9]   </pre>

Define playbook loops correctly

Use playbook loops only where needed. Loops are needed when certain actions have to be performed on specific pairs of data.

Correct Method Example	Incorrect Method Example
<p>Either use filters and transformers or loop through each separate indicator to verify they're creating the correct relationships.</p>	<p>A user has a playbook that creates relationships for multiple indicator types. All indicator types and malware families are in their <code>\$(inputs.Domain)</code> and <code>\$(inputs.MFam)</code> playbook inputs.</p> <p>The user wrongly assumes that when creating the relationships, the correct malware families in <code>\$(inputs.MFam)</code> correspond to the correct domains in <code>\$(inputs.Domain)</code>.</p> <pre> graph TD     A1[IA present? #19] -- NO --&gt; B1[Domain for IA #2]     A2[Malware Family present? #41] -- NO --&gt; B2[Domain for Malware Family #45]     B1 --&gt; C[...]     B2 --&gt; C   </pre>

Best practices for optimizing playbook design and performance

In order to minimize your case response time and make sure the system runs optimally, it's important to follow design and performance guidelines.

Use latest playbook and script versions

#### Playbooks

When returning to work on a playbook after a break, verify you're working on the latest version. Reattach the playbook if it's detached, and update it to ensure you're not editing an older version and introducing regressions. If you don't want to reattach your playbook, or you're still working on your custom version, we recommend reviewing the release notes to see what changes were made to the out-of-the-box playbook and copying those changes to your version.

#### NOTE:

If you reattach a detached playbook, any customizations you have made to the playbook will be overwritten when the playbook updates to the current version.

#### Scripts

Update scripts and integration commands in playbook tasks to their most current version. Scripts that have updates or are deprecated are designated by a yellow triangle.





Break up large playbooks into sub-playbooks

If a playbook has more than thirty tasks, consider breaking the tasks into multiple sub-playbooks. Sub-playbooks can be reused, managed easily when upgrading, and they make it easier to follow the main playbook.

Sub-playbooks are playbooks that are used from within a parent playbook, as building blocks. The parent playbook is the main playbook that runs on the investigation, and each sub-playbook has a specific goal/responsibility.

- Parent playbooks usually have a `closeInvestigation` task at the end because they are the main playbook for that issue.
- Parent playbooks usually contain inputs that are passed down to sub-playbooks. Certain `True/False` flags may come from the parent playbook inputs.

Remove unused playbook tasks

For production playbooks, remove playbook tasks that are not connected to the playbook workflow.

Set the playbook to run in quiet mode

Run playbooks in quiet mode to reduce the issue number size and execute playbooks faster.

For playbooks running in jobs, indicator enrichment should be done in quiet mode.

Only extract indicators when needed

When indicator extraction is enabled for a playbook task, the task by default tries to extract all indicator types from the task Results. (The Results entry is the information printed to the War Room, not the outputs of the task). Extracting all indicator types can slow down the playbook, so it is important to only extract indicators as needed. For example, for the ParseEmailFilesV2 script which prints email information to the War Room, extraction should be enabled in order to extract email addresses, URLs, and other indicators. However, if your task runs the Sleep script, there is no point in extracting indicators.

Set the Indicator Extraction mode to None in the playbook task Advanced tab.

Use retries

Retries help ensure smoother playbook execution and more efficient progress tracking.

Use retries when a task might temporarily fail but is expected to succeed later. This helps handle issues like network glitches, service downtime, or rate limits by retrying the task again after a short wait.

#### **NOTE:**

Retries are not supported for data collection tasks that have errors sending emails (indicated by a server timeout). This is because retries only work on automation execution failures, not on email delivery issues.

Use polling

Use polling to monitor a process or condition over time, especially when waiting for a specific outcome before proceeding such as waiting for an asynchronous task to complete. It periodically checks if a required condition is complete, ensuring your playbook moves forward only when ready. Common uses include waiting for a job to finish or a system to reach a certain state.

Minimize disk usage, CPU usage, and API calls

Consider the following:

- Do I need to do this action in multiple tasks?
- Can these tasks run in parallel instead of synchronously?
- Where applicable, am I setting realistic timeouts, search windows, intervals?
- Can I consolidate the API calls into one call? If not, can an integration enhancement solve this by accepting arrays as input instead of running multiple times for each input?
- Am I unnecessarily storing the same data twice? Do I have the data I need already stored?
- Where applicable, can I run this playbook without a loop?
- What extractions are running in my issue?

Get data from XQL datasets



You can leverage XQL for flexible and adjustable playbook and script tracking to provide performance and execution data. The following datasets are available for querying and dashboards:

- playbook\_tasks: Data about task executions within playbooks.
- playbook\_runs: Data about playbook runs and statuses.
- scripts\_and\_commands\_metrics: Data about scripts and commands used in playbook tasks.

### 3.8.5 | Create an automation rule

#### Abstract

Learn how to create an automation rule for an issue.

Automation rules allow users to automatically respond to events by defining trigger conditions and desired actions to perform once the condition is met.

#### **IMPORTANT:**

Rules are evaluated in order, and only the first rule that matches the trigger conditions is executed.

The rules consist of three parts: WHEN, IF, and THEN.

- WHEN: Stands for the trigger type, for instance, issue, case, or audit log. WHEN is set to Issue is created.
- IF: Stands for the conditions that need to be met for the rule to run.
- THEN: The action that the user wants to perform: playbook or Quick action.

In the Automation Rules page, you can create or edit an automation rule, use recommended automation rules, edit a playbook, and change the order of priority. You can also delete or disable/enable an automation rule. When you disable an automation rule, the automation does not run for the selected condition.

#### **NOTE:**

You can also define the conditions that trigger a specific playbook in the playbook editor. For more information, see Task 2. Configure playbook settings

#### Create or edit an automation rule

Create an automation rule for issues where conditions from the automation rule are met, so that the automation, whether it is a Quick Action or a playbook, automatically runs.

1. Go Investigation & Response → Automation → Automation Rules.
2. Click Add Automation Rule or right-click a rule, select Edit rule, or click the edit button.
3. Define the rule name and conditions:

1. Enter a rule name and set the rule status.

2. Under Rule Conditions:

- For If, click +Add Condition and from the Issues table, use the filter to set the criteria for the rule, and then click Save.  
For example, filter the field Severity, and then select the value Critical. The Issues table returns all issues where the severity=critical.
- For Then, click +Add Automation and from the Select Automations window, select the action you want to run.

You can search for the action or select a Quick Actions or a Playbook from the Org Playbooks or from the Playbook Catalog.

#### **NOTE:**

Quick Actions, by default, run using all available integration instances that contain the command. When selecting a Quick Action for an automation rule, you can instead choose one specific integration instance to use.

Click  to view the description and the tasks of the playbook.

For example, for the IF condition where severity=critical, select the Quick Action - Create Jira Ticket. The automation rule is triggered when a critical severity issue is detected, which then runs the selected automation, the Quick Action - Create Jira Ticket.

For more information on Quick Actions, see Quick Actions.

For more information on Playbooks, see Manage playbooks.

3. Save the automation rule.

#### Add a recommended automation rule

You can add automation rules recommended by Cortex AgentiX.



1. Go to Investigation & Response â†’ Automation â†’ Automation Rules.
2. Click View Recommendations.
3. In the Automation Rule Recommendations table, view and select the required recommended automation rules to add to the Automation Rules table.  
For playbooks, you can click the playbook name to preview. For Quick Actions, you can view the description and available parameters.
4. Click Add Selected rules.
5. Verify the order of the automation rule and change the order (if required),
6. Save the changes to the Automation Rules table.

After you create an automation rule, the rule is added to the Automation Rules table. In the Automation Rules table, you can do the following:

- Set the priority of the automation rules, so when an issue is created, the first rule takes priority, then the second, third, etc. Only the first matching rule is executed.

New rules created manually are added to the bottom of the table.

- View details of the automation rules that have been created.

By default, you can see the condition, automation, and the creation dates and source. You can add columns and filters as required. To edit, disable, or delete an automation rule, right-click on the rule.

#### Scope-based access control for automation rules

Automation rules support SBAC (scope-based access control). The following parameters are considered when editing a rule:

- If Scope-Based Access Control (SBAC) is enabled and Endpoint Scoping Mode is set to restrictive mode, you can edit an automation rule if you are scoped to all tags in the rule.
- If Scope-Based Access Control (SBAC) is enabled and Endpoint Scoping Mode is set to permissive mode, you can edit an automation rule if you are scoped to at least one tag listed in the rule.
- As a scoped user who has editing permissions to a rule, you can change the order among other rules that are locked.
- If a rule was added when set to restrictive mode, and then changed to permissive (or vice versa), you will only have view permissions.

### 3.8.6 | AI Prompts

On the AI Prompts page (Investigation â†’ Response â†’ Automation â†’ AI Prompts), you can view, edit, and manually create prompts.

#### 3.8.6.1 | AI prompts role-based access control

##### Abstract

Configure permissions to access AI prompts.

Instance and Account admins have full control over the permissions and access that users have to AI prompts. Cortex AgentX uses role-based access control (RBAC) to manage access to the AI prompts library, as well as access to create, edit and delete prompts in the prompts library and in the playbook editor.

By default, Instance and Account admins have full view/edit permissions enabled. When editing or creating other roles, in the CORTEX AGENTIC ASSISTANT section, you can enable AI Prompts. If you enable the feature for a role, the user can view prompts in the AI prompts library. In addition, after enabling AI Prompts, you can select the following:

Permission	Description
Manage prompts library	When selected, the user role can create, edit, and delete prompts in the AI prompts library.
Manage prompts in playbook editor	When selected, the user role can create and edit AI prompts in the playbook editor.

#### 3.8.6.2 | Use existing prompts

##### Abstract

Edit prompts to use in playbooks and run in the War Room.



Using an existing prompt allows you to quickly achieve reliable results by leveraging proven, pre-built instructions instead of starting from scratch. You can access the existing prompts from the Prompts Library, a centralized repository that helps you create, search, and edit your AI prompts. It enables turning prompts into reusable assets that can be shared across your organization and bring repeatability and control to your AI operations. The Prompts Library provides a dedicated space for organizing prompts across all your playbooks or for registering them as Actions and assigning them to Agents, and is particularly useful for managing long and complex prompts.

1. Navigate to Investigation & Response â“ Automation â“ AI Prompts and in the Prompts Library search for the prompt you want to use.



- Use free text in the search box to find an existing prompt. From the Basic dropdown, you can search for a prompt by Basic (name and tag), Name, or Tag.
- You can search for an exact match of the prompt name by putting quotation marks around the search text. For example, searching for "**VulnerabilityReportSummary**" returns the prompt with that name. You can search for more than one exact match by including the logical operator "or" in between your search texts in quotation marks. For example, searching for "**IssueSummaryAndRemediation**" or "**VulnerabilityReportSummary**" returns the two prompts with those names. Wildcards are not supported in free text search.
- You can sort the prompts in the library alphabetically, by modified date, by system, or custom, and you can filter for disabled or deprecated prompts.
- The Prompt Helper also provides a list of prompt writing tips, including:

Be clear and specific

Tell the AI exactly what you need.

Imagine you're asking a new team member for help — the more precise you are, the better they can assist. The same goes for our AI!

- What to do: Instead of vague questions like "Tell me about malware," try to be very specific. Think about:
  - The goal: What do you want to achieve? (for example, "Summarize," "Identify," "Explain," "Generate ideas")
  - The topic: What is the subject? (for example, "Phishing emails," "Vulnerability reports," "Security policies")
  - Any details: What specific information is important? (for example, "From last week's incidents," "For non-technical executives," "Highlighting critical threats")
- Examples:
  - Bad prompt: "Tell me about that virus \${VirusName}."
  - Good prompt: "Analyze the attached malware report from \${Path} and summarize the key indicators of compromise (IOCs) for our incident response team."

Provide context and background

Give the AI the full picture.

Our AI doesn't know everything about your specific situation. Giving it background information helps it understand the "why" behind your request.

- What to do: Include relevant details that help the AI understand the situation or your specific needs.
  - Role: Tell the AI to act as a specific persona (for example, "Act as a security analyst," "You are a CISO," "As a technical writer"). This helps it tailor its language and focus.
  - Audience: Who is the information for? (for example, "For a technical audience," "For a board meeting," "For a general user"). This influences the complexity and depth of the response.
  - Key Information: What specific data points or previous steps are relevant? (for example, "Based on the recent network scan results," "Considering the new compliance regulations").
- Examples:
  - Bad prompt: "Write a report."
  - Good prompt: "You are a cybersecurity consultant. Write a brief executive summary report for our CEO detailing the top three critical vulnerabilities identified in our recent penetration test report from \${Path} and suggest immediate actions."

Ask for the desired format

Guide the AI's output structure.

If you have a specific way you want the information presented, tell the AI upfront. This saves you time on reformatting.



- What to do: Clearly state how you want the AI's response to be structured.
  - Lists: "Provide a bulleted list of..." or "Give me 5 key points."
  - Tables: "Create a table with columns for [X], [Y], and [Z]."
  - Summaries/reports: "Generate a concise summary," "Draft a formal report," or "Write a brief email."
  - Length: "Keep it under 200 words," or "Provide a detailed analysis."
- Examples:
  - Bad Prompt: "What are the latest threats?"
  - "Good Prompt: "List the top 5 emerging cyber threats relevant to financial services, with a brief explanation for each, presented as a bulleted list."

#### Few-shot prompting

Use few-shot prompting when you need the AI prompt to learn a new pattern or format quickly without extensive fine-tuning, especially for tasks with limited data.

- What to do: Provide several examples of the desired input and output to guide the AI's response.
- Examples of good prompts:

"You are a SOC analyst that needs to enrich CVE \${CVEId} , use the following structure:"

Sample structures:

- CVE Description: Apache Struts 2.5.x before 2.5.14, 2.3.x before 2.3.34, and 2.x.x before 2.3.x.x.x allows remote attackers to execute arbitrary code via a crafted Content-Type header.
- CVSS:9.8 (Critical)Impact: Remote Code Execution (RCE), potential for complete system compromise, data theft, and denial of service. Affects web applications built with Apache Struts, widely used in enterprise environments.
- Risk Score: 10/10 - Extremely High. Exploitability is high due to public exploits and widespread usage of the affected software.
- CVE Description: Microsoft Windows MSHTML Remote Code Execution Vulnerability. This vulnerability exists in the way MSHTML engine handles specially crafted files. An attacker could host a specially crafted website or send a specially crafted document that, when opened, could allow remote code execution.
- CVSS:8.8 (High)Impact: Remote Code Execution (RCE), arbitrary code execution in the context of the current user. Affects all Windows versions. Could lead to system compromise and data exfiltration. Often exploited via phishing campaigns.
- Risk Score: 9/10 - Very High. Widespread target, often exploited through user interaction, making it a common attack vector.

2. Click Edit. If the prompt you want to use is locked, click  and then select Duplicate Prompt.

System prompts, are by default locked, which means they are not editable. To edit a system prompt, you need to make a copy.

3. Edit the prompt and settings as needed.

For details about prompt settings, see Create a prompt.

4. (Recommended) Click Optimize to optimize and improve the prompt edits based on predefined system guidelines.

The suggested prompt replaces the existing one. You can undo the optimization if needed.

5. Save the prompt version.

6. (Recommended) Click Test to validate your prompt.

1. In the Arguments section, provide values for any inputs your prompt requires. These inputs are used to simulate how the prompt will behave in a live playbook, or how the prompt as an Action for an Agent will run as part of an executed plan.

You can add input values manually.

2. Click Run.

The tests are executed in a Playground environment. Review the output generated by the AI to validate the prompt's behavior and ensure it produces the expected results. The output is typically a text summary or another structured format that you have defined.

In each run result, you can take the following actions:



Action	Description
Mark as note	Marks the entry as a note, which can help you understand why certain action was taken and assist future decisions. When marked as a note, it is highlighted, so you can easily find it in the War Room or the Issue Overview tab.
View artifact in new tab	Opens a new tab for the artifact.
Download artifact	Downloads the run details to a text file, including the AI task name, the prompt name, user name and password, and the result.
Add tags	Add any relevant tags to use that help you find relevant information.



7. (Optional) Click  and select Register new Action to register the prompt as an Action and make it available for Agents. For more information, see Manage actions.

8. (Optional) Add the prompt to a playbook.

1. Edit or create a playbook.

2. In the playbook editor, expand the Task Library and select AI Prompts.

The System tab contains system prompts, and the Custom tab contains custom prompts.

3. Select the relevant prompt and drag it onto the playbook editor.

The Task Details pane opens for the prompt. You can view system prompt details, and you can view and edit custom prompt details.

4. Click OK.

The prompt appears in the playbook editor.

### 3.8.6.3 | Create a prompt

#### Abstract

Create or edit an out-of-the-box prompt, including detach and attach and automation settings.

Creating a prompt enables you to turn your own custom requests into reusable, shareable AI prompts in playbooks and as Actions for Agents.

1. Navigate to Investigation & Response → Automation → AI Prompts and click + New Prompt.

2. Add an identifying name for the prompt.

3. Save the prompt.

4. Enter prompt settings.

#### Basic settings

Define the relevant basic prompt parameters.

Parameter	Description
Name	An identifying name for the prompt.



Parameter	Description
Description	<p>A meaningful description of the prompt. If you want to register the prompt as an action, make the description as detailed as possible.</p> <p>For example, for the Cortex - Blocklist Files action, the description is:</p> <p>Blocklists the specified SHA256 file hashes in Cortex by adding them to Cortex's blocklist. Skips any that already exist in Cortex's allowlist or blocklist. Optionally returns detailed results with counts of added and skipped hashes.</p>
Tags	<p>Predefined prompt identifiers.</p> <p>For example, if a prompt is intended for phishing, tagging it with the phishing tag helps organize, classify, and manage the prompt among other prompts.</p>

#### Advanced settings

Define settings to help optimize your prompt.

Parameter	Description
Temperature	Temperature enables customizing for pinpoint accuracy or diverse outputs by controlling the randomness of AI responses. The value must be between 0 and 2. Lower values (0-0.3) produce more focused, deterministic responses. Higher values (0.7-2) produce more creative, varied responses. You can set the value by entering a number or by adjusting the number on a slider.
Max Output Tokens	Max Output Tokens ensure responses adhere to specific length constraints by setting the maximum number of tokens the AI model can generate in response. Default is 2500 tokens. You can set the value by entering a number or by adjusting the number on a slider.

5. Enter the prompt in the Prompt pane as well as any relevant inputs.

Inputs can be set with either context path or specific value. You can choose whether the input is a variable using \${<input name>}.

The Prompt Helper also provides a list of prompt writing tips, including:

Be clear and specific

Tell the AI exactly what you need.

Imagine you're asking a new team member for help — the more precise you are, the better they can assist. The same goes for our AI!

- What to do: Instead of vague questions like "Tell me about malware," try to be very specific. Think about:
  - The goal: What do you want to achieve? (for example, "Summarize," "Identify," "Explain," "Generate ideas")
  - The topic: What is the subject? (for example, "Phishing emails," "Vulnerability reports," "Security policies")
  - Any details: What specific information is important? (for example, "From last week's incidents," "For non-technical executives," "Highlighting critical threats")
- Examples:
  - Bad prompt: "Tell me about that virus \${VirusName}."
  - Good prompt: "Analyze the attached malware report from \${Path} and summarize the key indicators of compromise (IOCs) for our incident response team."

Provide context and background

Give the AI the full picture.

Our AI doesn't know everything about your specific situation. Giving it background information helps it understand the "why" behind your request.



- What to do: Include relevant details that help the AI understand the situation or your specific needs.
  - Role: Tell the AI to act as a specific persona (for example, "Act as a security analyst," "You are a CISO," "As a technical writer"). This helps it tailor its language and focus.
  - Audience: Who is the information for? (for example, "For a technical audience," "For a board meeting," "For a general user"). This influences the complexity and depth of the response.
  - Key Information: What specific data points or previous steps are relevant? (for example, "Based on the recent network scan results," "Considering the new compliance regulations").

- Examples:

- Bad prompt: "Write a report."
- Good prompt: "You are a cybersecurity consultant. Write a brief executive summary report for our CEO detailing the top three critical vulnerabilities identified in our recent penetration test report from \${Path} and suggest immediate actions."

Ask for the desired format

Guide the AI's output structure.

If you have a specific way you want the information presented, tell the AI upfront. This saves you time on reformatting.

- What to do: Clearly state how you want the AI's response to be structured.
  - Lists: "Provide a bulleted list of..." or "Give me 5 key points."
  - Tables: "Create a table with columns for [X], [Y], and [Z]."
  - Summaries/reports: "Generate a concise summary," "Draft a formal report," or "Write a brief email."
  - Length: "Keep it under 200 words," or "Provide a detailed analysis."
- Examples:
  - Bad Prompt: "What are the latest threats?"
  - "Good Prompt: "List the top 5 emerging cyber threats relevant to financial services, with a brief explanation for each, presented as a bulleted list."

Few-shot prompting

Use few-shot prompting when you need the AI prompt task to learn a new pattern or format quickly without extensive fine-tuning, especially for tasks with limited data.

- What to do: Provide several examples of the desired input and output to guide the AI's response.
- Examples of good prompts:

"You are a SOC analyst that needs to enrich CVE \${CVEId} , use the following structure:"

Sample structures:

- CVE Description: Apache Struts 2.5.x before 2.5.14, 2.3.x before 2.3.34, and 2.x.x before 2.3.x.x.x allows remote attackers to execute arbitrary code via a crafted Content-Type header.
- CVSS:9.8 (Critical)Impact: Remote Code Execution (RCE), potential for complete system compromise, data theft, and denial of service. Affects web applications built with Apache Struts, widely used in enterprise environments.
- Risk Score: 10/10 - Extremely High. Exploitability is high due to public exploits and widespread usage of the affected software.
- CVE Description: Microsoft Windows MSHTML Remote Code Execution Vulnerability. This vulnerability exists in the way MSHTML engine handles specially crafted files. An attacker could host a specially crafted website or send a specially crafted document that, when opened, could allow remote code execution.
- CVSS:8.8 (High)Impact: Remote Code Execution (RCE), arbitrary code execution in the context of the current user. Affects all Windows versions. Could lead to system compromise and data exfiltration. Often exploited via phishing campaigns.
- Risk Score: 9/10 - Very High. Widespread target, often exploited through user interaction, making it a common attack vector.

## 6. (Optional) Click Optimize to optimize and improve the prompt based on predefined system guidelines.

The suggested prompt replaces the existing one. You can undo the optimization if needed.

## 7. Click Save Version to save the prompt version.

## 8. (Recommended) Click Test to validate your prompt.



1. In the Arguments section, provide values for any inputs your prompt requires. These inputs are used to simulate how the prompt will behave in a live playbook, or how the prompt as an Action for an Agent will run as part of an executed plan.

You can add input values manually.

2. Click Run.

The tests are executed in a Playground environment. Review the output generated by the AI to validate the prompt's behavior and ensure it produces the expected results. The output is typically a text summary or another structured format that you have defined.



9. (Optional) Click and select Register new Action to register the prompt as an Action and make it available for Agents. For more information, see [Manage actions](#).

10. (Optional) Add the prompt as an AI prompt task to a playbook.

1. Edit or create a playbook.

2. In the playbook editor, expand the Task Library and select AI Prompts.

The System tab contains system AI prompt tasks, and the Custom tab contains custom AI prompt tasks.

3. Select the relevant AI prompt task and drag it onto the playbook editor.

The Task Details pane opens with the prompt appearing in the Prompt field.

4. Click OK.

The prompt appears in the playbook editor.

### 3.8.7 | Scripts

On the Scripts page (Investigation â– Response â– Automation â– Scripts), you can view, edit, and create scripts in JavaScript, Python, or PowerShell.

When creating a script, you can access all APIs, including cases and investigations, and share data in the War Room. Scripts can receive and access arguments and can be password-protected.

#### PREREQUISITE:

To provide scripts access, ensure the Scripts RBAC permission is set to View or View/Edit.

To completely restrict scripts access, first set Playbooks and Playground RBAC permissions to None and then set the Scripts permission to None. Conversely, to provide access to playbooks or the Playground, the Scripts permission must first be set to View or View/Edit (and then set the Playbooks or Playground permissions).

Automation Engineer agent

Use the AI-powered Automation Engineer agent to simplify Python script creation and management through an intuitive, interactive experience. It enables you to generate, modify, and query automation scripts with the Agentic Assistant natural language chat prompt. For example, within the chat, you can ask the agent to explain the specific script currently open or pose general technical questions regarding script logic and the AgentiX SDK.

For more details about using the Automation Engineer agent, see [Use the Automation Engineer agent to accelerate script development and deployment](#).

#### 3.8.7.1 | Use existing scripts

Abstract

Edit scripts to use in playbooks and run in the War Room.

Using or modifying an existing script enables you to quickly leverage proven functionality and save significant time and effort developing a new script from scratch.

For example, you can use scripts from the Base and Common Scripts content packs that provide basic and reusable functions that can streamline your playbook development.

Common scripts

Cortex AgentiX comes out-of-the-box with several common scripts that can be used in playbooks and commands (from the War Room), the majority of which are contained in the Base and Common Scripts content packs.

The Base content pack is a core pack that helps you get started and includes scripts that can be used in other JavaScript, Python, and PowerShell scripts. The Common Scripts content pack includes scripts that are commonly used, such as EmailReputation, RunDockerCommand, and ConvertXMLToJson.

Common Scripts contain code (such as functions and variables) that can be used across scripts and can be embedded when writing your scripts and integrations. Common Scripts are reusable modules or functions that provide additional functionality and capabilities to interact with APIs. Instead of duplicating code across multiple scripts or integrations, developers can create common scripts containing commonly used API interactions, such as



authentication, data retrieval, or data manipulation. For example, in the **CommonServer** script, the `tableToMarkdown` function takes a JSON and transforms it into markdown. You can call this function from integrations and scripts that you author.

On the Scripts page, you can view/edit common scripts such as:

- CommonServer

The CommonServer script contains JavaScript functions and variables that can be used when writing your scripts and integrations.

The script contains nearly 200 functions/variables, such as `tableToMarkdown`, `closeInvestigation`, and `SetSeverity`.

You can copy the script and add new functions/variables, or add your functions to the CommonUserServer script. You can also use your scripts to override the existing scripts in the CommonServer script.

- CommonServerPython

The CommonServerPython script contains Python functions that can be used when writing your scripts and integrations.

The script contains over 400 functions, such as `appendContext`, `vtCountPositives` (which counts the number of detected URLs in the War Room entry), and `datetime_to_string`, (which converts a DateTime object into a string).

You can copy the script and add new functions/variables, or add your functions to the CommonServerUserPython script. You can also use your scripts to override the existing scripts in the CommonServerPython script.

- CommonServerPowerShell

The CommonServerPowerShell script contains PowerShell arguments/functions that can be used when writing your scripts and integrations.

The script contains many arguments/functions, such as `SetIntegrationContext`, `Write-HostToLog` (which writes to the demisto.log), and `ReturnOutputs` (which returns results to the user more intuitively).

You can copy the script and add new arguments/functions or add your own to the CommonServerUserPowerShell script. You can also use your scripts to override the existing scripts in the CommonServerPowerShell script.

1. Navigate to Investigation & Response → Automation → Scripts and in the Scripts Library search for the script you want to use.

- Use the free text in the search box to find an existing script. From the search drop-down, you can:
  - Perform a basic search by Basic (name and tag), Name, or Tag.
  - Perform an advanced search for specific words In Script or Everywhere (including the script name and tags).
- You can search for an exact match of the script name by putting quotation marks around the search text. For example, searching for "`AddKeyToList`" returns the script with that name. You can search for more than one exact match by including the logical operator "or" in between your search texts in quotation marks. For example, searching for "`AnalyzeTimestampIntervals`" or "`AddKeyToList`" returns the two scripts with those names. Wildcards are not supported in free text search.
- You can sort the scripts in the library alphabetically, by modified date, by system, or custom, and you can filter for disabled or deprecated scripts.
- The Script Helper also provides a list of available alphabetically ordered commands and scripts.

2. Click Edit. If the script you want to use is locked, you first need to duplicate it.

If a script is installed from a content pack, by default, the script is locked, which means that it is not editable.

3. In the Agentic Assistant pane, start a conversation with the Automation Engineer agent to edit the script, or manually edit the script code and define the script settings.

For more information, see Use the Automation Engineer agent to accelerate script development and deployment. For details about script settings, see Create a script.

4. Save the script version.

5. (Recommended) Validate your script.

1. Click Test.
2. In the Arguments section, provide values for any inputs your prompt requires. These inputs are used to simulate how the script will behave in a live playbook, or how the script registered as an Action and assigned to an Agent will run as part of an executed plan.

You can add input values manually.

3. Click Run.

The scripts are executed in the Playground. Review the output generated by the script to validate its behavior and ensure it produces the expected results. The output is typically a text summary or another structured format that you have defined.

In each run result, you can take the following actions:



Action	Description
Edit	Edit the entry, mark it as a note, preview it, or delete it.
Mark as note	Marks the entry as a note, which can help you understand why certain action was taken and assist future decisions. When marked as a note, it is highlighted, so you can easily find it in the War Room or the Issue Overview tab.
View artifact in new tab	Opens a new tab for the artifact.
Add tags	Add any relevant tags to use that help you find relevant information.



6. (Optional) Click next to the Edit button and select Register new Action to register the script as an action and make it available for agents. For more information, see [Manage actions](#).

### 3.8.7.2 | Create a script

#### Abstract

Create or edit an out-of-the-box script, including detach and attach and automation settings.

Creating custom scripts in Cortex AgentiX helps meet your organization's specific needs to automate repetitive tasks, streamline security operations, and make case response more efficient.

1. Navigate to Investigation & Response → Automation → Scripts and click New Script.
2. Add an identifying name for the script.
3. Click Save.
4. In the Agentic Assistant pane, start a conversation with the Automation Engineer agent to create the script, or manually create the script code and define the script settings.

For more information, see [Use the Automation Engineer agent to accelerate script development and deployment](#). For details about script settings, see [Create a script](#).

5. Save the script version.

6. (Recommended) Click Test to validate your script.

1. In the Arguments section, provide values for any inputs your prompt requires. These inputs are used to simulate how the script will behave in a live playbook, or how the script registered as an Action and assigned to an Agent will run as part of an executed plan.

You can add input values manually.

2. Click Run.

The tests are executed in a Playground environment. Review the output generated by the AI to validate the script's behavior and ensure it produces the expected results. The output is typically a text summary or another structured format that you have defined.

#### TIP:

If there is an error, you can copy the error message from the test result into the Agentic Assistant prompt and ask the Automation Engineer agent to correct the error.

In each run result, you can take the following actions:

Action	Description
Mark as note	Marks the entry as a note, which can help you understand why certain action was taken and assist future decisions. When marked as a note, it is highlighted, so you can easily find it in the War Room or the Issue Overview tab.



Action	Description
View artifact in new tab	Opens a new tab for the artifact.
Download artifact	Downloads the run details to a text file, including the AI task name, the script name, user name and password, and the result.
Add tags	Add any relevant tags to use that help you find relevant information.

7. (Optional) Click  and select Register new Action to register the script as an Action. For more information, see Manage actions.

**NOTE:**

- You can enable/disable a script in the Settings without having to duplicate the script.
- You can view recently modified or deleted scripts by clicking the version history for all scripts .

Basic script settings

Define the relevant Basic script parameters.

Parameter	Description
Name	An identifying name for the script.
Language type	Select the script language type. <b>IMPORTANT:</b> If you choose Python, from the Agentic Assistant you can use the Automation Engineer agent.
Description	A meaningful description of the script.
Tags	Predefined script identifiers. For example, if a script is intended for phishing, tagging it with the phishing tag helps organize, classify, and manage the script among other scripts. Organizations can also implement policies or restrictions based on tags associated with scripts. For example, they may restrict certain users from accessing or executing a script tagged for phishing.
Enabled	Whether the script is available for playbook tasks and indicator types, or to run in the CLI.

Arguments

You can create, edit, or delete arguments as required.

Parameter	Description
Argument	An identifying name.
Mandatory	Makes the argument mandatory.



Parameter	Description
Default	Makes the argument the default.
Sensitive	Hides the argument from being displayed in the UI and in logs.
Description	A meaningful description of the argument.
Default	The default value for the argument.
Is array	Specifies that the argument is an array.
List options	A comma-separated list of argument values.

You can create, edit, or delete outputs as required. Define the outputs according to types such as string, number, date, and Boolean. For more information, see Context and Outputs.

Parameter	Description
Context Path	A dot-notation representation of the path to access the Context. For example, <code>ThreatStream.Analysis.ReportID</code> .
Description	A short description of what the context path represents. For example, the ID of the report submitted to the sandbox.
Type	The value type of the context path, such as string, number, and date, enables Cortex AgentX to format the data correctly.

#### Script permissions

Parameter	Description
Password Protect	Enables you to add a password for the script, which will be required when running the script from the CLI.

#### Advanced

Parameter	Description
Timeout (seconds)	Time (in seconds) before the script times out. Default is 180.



Parameter	Description
Docker image name	<p>For Python scripts, this is the name of the Docker image to use for the script.</p> <p>Cortex AgentiX supports the following Python versions:</p> <ul style="list-style-type: none"> <li>• 2.7</li> <li>• 3.0 and later</li> </ul> <p>You can change the Docker image.</p> <p>The default Docker image that Cortex AgentiX uses is <code>demisto/python3</code>, but you can use other Docker images.</p>
Run on a separate container	Runs the script on a separate container.

Depends on commands

You can set the commands that the script depends on directly from these settings. You still have the option to set the dependencies in the script YAML file.

Edit existing code or create new code

Modify parameters, logic, or integrations within a script to adapt it to specific use cases, optimize performance, and address evolving security needs without starting from scratch.

The Script Helper provides a list of available alphabetically ordered commands and scripts.

### 3.8.7.3 | Use the Automation Engineer agent to accelerate script development and deployment

#### Abstract

Use the AI-powered Automation Engineer agent to create, modify, explore, and understand scripts.

The **Automation Engineer** agent is a conversational AI that simplifies Python script creation and management through an intuitive, interactive experience. It enables you to generate, query, iterate, and refine automation scripts with the AgentiX natural language chat prompt.

Automation Engineer agent scripting capabilities include:

- Initial code generation: Generate a full script from a simple prompt. For example, "Generate a script to change the verdict of a given indicator based on user input, including documentation notes and debug messages."
- Existing script modification: For example, "Add a check to ensure the indicator exists before proceeding with the verdict change, and return an error if it does not."
- Iterative bug fixing: For example, "Provide specific error messages for the agent to analyze and automatically repair."
- API compatibility updates: For example, "Detect and replace deprecated API calls across an entire script."
- Logic simplification: Ask the agent to refactor complex code to be more readable or to remove redundant conditionals for simple lookups.
- Script explanation: Ask the agent questions about system or custom scripts, including asking how the script works. You can ask the agent to explain the specific script currently open or pose general technical questions regarding script logic and the AgentiX SDK.
- Input and error validation: Enhance script robustness by asking the agent to add specific try/except blocks or validate that inputs like username are not empty.
- SDK and command guidance: Ask the agent for technical details on using the AgentiX SDK or the proper syntax for running commands within a script.

When you download or update content packs, the new or updated scripts are immediately ready for the Automation Engineer agent to recommend and use.

#### NOTE:

The Automation Engineer agent is available for users with script editing permissions.

How to use the Automation Engineer agent

1. From the Investigation & Response → Automation → Scripts page, either choose an existing script or create a new script.



For a new script:

1. Click + New Script, give the script a name, and click Save.



2. Click . The Agentic Assistant pane opens with the Automation Engineer agent automatically selected.

For an existing script:

1. In the Scripts Library, search for the script you want to use.

#### Search tips

- Use the free text in the search box to find an existing script. From the search drop-down, you can:
  - Perform a basic search by Basic (name and tag), Name, or Tag.
  - Perform an advanced search for specific words In Script or Everywhere (including the script name and tags).
- You can search for an exact match of the script name by putting quotation marks around the search text. For example, searching for "AddKeyToList" returns the script with that name. You can search for more than one exact match by including the logical operator "or" in between your search texts in quotation marks. For example, searching for "AnalyzeTimestampIntervals" or "AddKeyToList" returns the two scripts with those names. Wildcards are not supported in free text search.
- You can sort the scripts in the library alphabetically, by modified date, by system, or custom, and you can filter for disabled or deprecated scripts.
- The Script Helper also provides a list of available alphabetically ordered commands and scripts.

2. Click Edit. The Agentic Assistant pane automatically opens with the Automation Engineer agent selected.

If a script is installed from a content pack, by default, the script is locked, which means that it is not editable. To edit a system script, you first need to duplicate it.

If you are in the middle of a chat with a different agent, you are prompted to start a new chat with the Automation Engineer agent.

If you start a chat with one script and switch to another script, you are prompted to start a new chat.

2. In the Agentic Assistant pane, enter a natural language prompt describing what you need the agent to do, including:

- Explain what the script does.
- Fix code errors: If there are errors in the script code, you can ask the Automation Engineer agent to suggest a correction.
- Add documentation notes to the script: Ask to include explanations and inline comments.
- Add arguments to the script: Define the inputs your script should accept. Each argument should include a name, type, whether it is required, and optionally a default value.

Examples:

- days ← number, optional, default: 3
- email ← string, required, email="soc@company.com"
- Add outputs to the script: Describe what the script should return to the context, for example, recentIncidentsSummary ← string, a human-readable summary of incidents.
- Include debug logging: Request to include contextual log messages in the script.
- Include error handling: Ask to include try/except blocks with informative error logs.

#### TIP:

- Use detailed prompts, for example, Get failed logins from last 24 hours and return as a table.
- Clearly define argument names, types, and default values.
- Mention if you expect the output in a specific format, such as a table, JSON, or plain text.

Example 34.

The following are sample prompts:

```
Fetch all open incidents from the last 3 days.  
Generate a summary table with ID, name, and severity.  
Email the summary to the given address.
```

```
Arguments:  
- days (number, default: 3)  
- email (string, required)
```



```
Output:  
- incidentsSummary (string): a human-readable table of incident details  
  
Explain what this script does  
  
How do I use the SDK to execute a search command?  
  
I got this error: <KeyError>: <userId>. Fix it.
```



3. Click or Enter to submit the prompt.

The Agentic Assistant then displays:

- The plan describing the steps the Automation Engineer agent took.
- A script preview card that includes the following details:
  - The script name.
  - The script revision number (#).
  - Script metadata: The number of lines, arguments, and outputs.
  - An expand icon that shows the new script code with the option to Use this revision.A small black circle with three vertical dots inside.
  - that includes Use this revision or Copy code.

The first line of the generated script indicates it was generated by AgentiX, with the date time of the latest update.

4. Use natural language in the prompt to modify the generated script as needed.

Example 35.

For the script generated from the sample prompt above, enter the following modification to add sorting:

```
Modify the sorting behavior so that all 1s (threats) come before all 0s (safe events). Keep the rest of the script structure the same.  
1 / 1
```

#### NOTE:

If you make manual edits in the script and don't save the changes and you then modify the script with the Automation Engineer agent, you are prompted to confirm overwriting the manual edits.



5. (Optional) Access an earlier script revision by clicking and then Use this revision on the script preview card of the revision you want to use.

6. Continue modifying and submitting prompts until the script works as intended.

7. For a new script, click Save Version. For an existing script that was edited, click Use this revision and then Save Version.

8. (Recommended) Validate your script.

1. Click Test.

2. In the Arguments section, provide values for any inputs your prompt requires. These inputs are used to simulate how the script will behave in a live playbook, or how the script registered as an Action and assigned to an Agent will run as part of an executed plan.

You can add input values manually.

3. Click Run.

The scripts are executed in the Playground. Review the output generated by the script to validate its behavior and ensure it produces the expected results. The output is typically a text summary or another structured format that you have defined.

#### TIP:

If there is an error, you can copy the error message from the test result into the Agentic Assistant prompt and ask the Automation Engineer agent to correct the error.

In each run result, you can take the following actions:

Action	Description
Edit	Edit the entry, mark it as a note, preview it, or delete it.



Action	Description
Mark as note	<p>Marks the entry as a note, which can help you understand why certain action was taken and assist future decisions.</p> <p>When marked as a note, it is highlighted, so you can easily find it in the War Room or the Issue Overview tab.</p>
View artifact in new tab	Opens a new tab for the artifact.
Add tags	Add any relevant tags to use that help you find relevant information.

### 3.8.7.4 | Change the Docker image in an integration or script

#### Abstract

Use Docker to run Python scripts and integrations in a controlled environment in Cortex AgentiX.

Docker enables you to run scripts and integrations from an image in a controlled environment that isolates and safeguards the tenant. It also simplifies environment setup by packaging dependencies and configurations within an image, ensuring consistent execution across different systems. By default, Cortex AgentiX pulls images from the [Demisto](#) Docker image registry in GitHub, which are used in scripts and integrations as needed. Cortex AgentiX integrations and scripts have the relevant Docker image already selected. For example, the Rasterize integration uses the [demisto/python.3.3.11.9.1079](#) Docker image.

You may want to select a different Docker image for your integration or script. In Cortex AgentiX, you can select a different Docker image from a dropdown that is pulled from the [Demisto](#) Docker image registry. In GitHub, the dockerfiles-info branch contains information about each image to help you find one that is relevant.

#### NOTE:

You can access publicly available Docker images from the Cortex AgentiX tenant even if there is no external connection to the [Demisto](#) registry, for example, if due to firewall constraints, your engine cannot access the [Demisto](#) registry.

#### Change the Docker image for a script

1. Edit the script.
2. Under ADVANCED, in the Docker image name field, click X to clear the current selection and then select a Docker image name from the dropdown menu.

For more information about changing the Docker image for a script, see the Advanced tab in Create a script.

3. Save your changes.

#### Change the Docker image for an integration

1. Navigate to Settings → Data Sources & Integrations, find and select your integration and edit the integration's source.

For an out-of-the-box content pack integration, you first need to duplicate the integration to edit it.

2. In the Integration Settings, expand the Script section.
3. Click X to clear the current selection and select a Docker image name from the dropdown menu.

For more information about changing the Docker image, see the Advanced tab in Create a script.

4. Save your changes.

### 3.8.8 | Context data

#### Abstract

Use context data to assist with the investigation and remediation process.

Context data is a map (dictionary) that stores structured data related to an issue, including issue fields and automations data. You can use context data to pass data between playbook tasks, and create scripts that map data into case and issue fields.



### 3.8.8.1 | Issue context data



To see context data for an issue, open the issue card and click the Issue Context Data icon

Consider the following information when working with context data:

- When an issue is created, the issue field data is stored under the `issue` key in the context data. When an investigation is opened and commands are run, the data returned from those commands is stored outside of the main `issue` key.
- Issue context data is split into two tabs. The Issue tab contains the context data from the issue fields and the commands run on the issue. The Case tab contains the parent case fields and other case data. None of this data is added to the context data for the parent case unless you add it.
- You can add keys and values to the context data. This is useful when developing playbooks, and other automations. For more information, see Add context data to an issue.
- When running automations on an issue, the issue can access context data from its parent case; however, it cannot access context data from other issues. If you want to use context data from other issues, add it to the parent case.

### 3.8.8.2 | Case context data

Context data is written to issues and not to cases. Therefore, the case context might be empty unless you previously added context data to the case.

To see context data for a case, open a case, click the Actions menu and select View context data.

Adding context data from issues to a parent case can help you with the following tasks:

- Remediation:** You can add context data from an issue, such as the issue status, actions, or ID, to its parent case's context data. This allows other playbooks to use the parent case context.  
For example, if you have multiple issues in a case, you can add context data from each of the issues to the parent case. You can then use the case context data in playbooks, and avoid running duplicate actions on the issues.
- Case assignment:** You can see if an analyst has been assigned to the case or other issues.
- Insights at the case level:** For automation engineers, you can set responses based on characteristics in the case.

For more information, see Add context data to a case.

### 3.8.8.3 | Search context data

Abstract

Use Query to search for specific items in the context data of a case or issue.

You can use Query to search within the context data JSON for specific items and expand nested keys. Open the context data panel for an issue or case, as explained in Issue context data or Case context data, and type in the Search field.

Example context:

```
{
  "HelloWorld": {
    "Alerts": [
      {
        "name": "Example 1",
        "alert_status": "ACTIVE"
      },
      {
        "name": "Example 2",
        "alert_status": "CLOSED"
      },
      {
        "name": "Example 3",
        "alert_status": "ACTIVE"
      }
    ]
  }
}
```

Search examples:



- `${c}`  finds the value of the object `c`.
- `${HelloWorld.Alert(val.name == 'Example 1')}`  shows the full object for the alert named "Example 1", as stored in the context data.
- `${HelloWorld.Alert(val.alert_status === "ACTIVE")}`  shows the full object for all alerts in context with status "ACTIVE".
- `${HelloWorld.Alert(val.alert_status == 'ACTIVE').name}`  fetches the `HelloWorld.Alert.name` of all alerts in context with status "ACTIVE".

### 3.8.8.4 | Add context data to an issue

#### Abstract

Use a script, command, or playbook to add context data to an issue to be used in playbooks or other automations.

You can add keys and values to an issue's context data to be used in playbooks or other automations.

To add context data to an issue, run the `Set` command in CLI, in a script, or in a playbook task. The Set command enables you to set a value under a specific key. For more information about the Set command, see Set.

#### Use the CLI

Run the `!Set` command in the issue War Room.

1. Open an issue and select the War Room tab.
2. Run the `!Set` command.

#### Example 36. Example

The following example adds the key and value `hello:world` to the issue context data.

```
!set key="hello" value="world"
```

#### Use a script

In the JSON file, add `Set` to the `demisto.executeCommand` key.

#### Example 37. Example

The following example adds the key and value `hello:world` to the issue context data.

```
demisto.executeCommand("Set", {"key": "hello", "value": "world"})
```

#### Use a playbook

Use the `Set` script in a standard task.

#### Example 38. Example

An issue's context data contains the following values:

```
{
  "Account": [
    {
      "firstName": "Bob",
      "lastName": "Jones"
    }
}
```

For an automation, you need to use the full name value. You can use the `Set` script to add a new `fullName` value to the JSON:



Task Type: Set

Set Full Name

Scripts: Set

Inputs Outputs Mapping Advanced Details On Error

key\* Account.fullName

value\* \${Account.firstName} \${Account.lastName}

Result:

```
{
  "Account": {
    "firstName": "Bob",
    "fullName": "Bob Jones"
    "lastName": "Jones",
  }
}
```

### 3.8.8.5 | Add context data to a case

#### Abstract

Use a script, command, or playbook to add context data to a case to be used in playbooks or other automations.

You can add keys and values to a case's context data to be used in playbooks or other automations. By default, context data is added to cases only. To run automations on a case, add context data to the case from its related issues.

To add context data to a case, run the `setParentIncidentContext` command in the CLI, in a script, or in a playbook task.

#### Use the CLI

Run the `!setParentIncidentContext` command in the issue War Room or the Case War Room.

#### **NOTE:**

If you run the command in the issue War Room, the data is added to the following places:

- The case context data.
- The issue context data under the case tab.

If you run the command in the Case War Room, the data is added to the case context data only.

Run the command in the issue War Room

1. Open an issue and select the War Room tab.
2. Run the `!setParentIncidentContext` command.

#### Example 39. Example

The following example adds the key and value `hello:world` to the case and issue context data.

```
!setParentIncidentContext key="hello" value="world"
```

Run the command in the case War Room

1. Open a case and switch to the Detailed view.
2. Select the Case War Room tab.
3. Run the `!setParentIncidentContext` command.



#### Example 40. Example

The following example adds the key and value `hello:world` to the case context data.

```
!setParentIncidentContext key="hello" value="world"
```

#### Use a script

In any script that runs in an issue, the data is written to the issue context data. If you want to add the data to the case context from your script, run the `setParentIncidentContext` using the `demisto.executeCommand` key, as follows:

```
demisto.executeCommand("setParentIncidentContext", {"key": "<key>", "value": "<value>"})
```

#### Example 41. Example

The following example creates a new key name `AuditID` with a `90210` value to your script.

```
demisto.executeCommand("setParentIncidentContext", {"key": "AuditID", "value": "90210"})
```

#### Use a playbook

When a playbook runs, the playbook data is written to the issue context data. To write the data to the parent case context data, use the `setParentIncidentContext` script in a standard task.

#### Example 42.

The following example adds the `TicketID` to the case context. To see a full use case that includes this standard task, see Use context data in a playbook.

The screenshot shows the 'Task Details' interface for a standard task named '#17 Set Ticket ID'. The task type is set to 'Standard'. The 'Script' field contains the command `setParentIncidentContext (Builtin)`. The 'Inputs' tab is active, showing the following configuration:

- key\***: `TicketID`
- value\***: `Ticket.Id`
- Get**: `Ticket.Id`
- Where**: `No filters applied`
- Transformers**: `No transformers applied`

#### 3.8.8.6 | Delete context data from a case

##### Abstract

You can delete context data from a case by running a command in the Case War Room or the issue War Room.

Run the `!deleteParentIncidentContext` command to delete all context data or a specific key in the Case War Room or issue War Room.

Use the issue War Room

1. Identify an issue and click to Investigate the issue.
2. In the issue investigation panel, select the War Room tab.
3. Run the `!deleteParentIncidentContext` command.

Use the case War Room



1. In the case investigation panel, select the Case War Room tab.

2. Run the `!deleteParentIncidentContext` command.

#### Example 43. Example

The following example deletes the key and value `hello:world` from the case or issue context.

```
!deleteParentIncidentContext key="hello" value="world"
```

#### 3.8.8.7 | Use context data in a playbook

##### Abstract

Learn how to use context data in playbook tasks, and how to update context data from a playbook.

In Cortex AgentiX you can use context data (from an issue or case) in playbooks, and you can use playbook tasks to update context data. You can:

- Use the information stored in the issue context data as task inputs and outputs in a playbook.
  - To access data that is stored in the issue context data, use the keyword `issue`.

#### Example 44.

To access a the `status` value in the issue context data, use the following syntax:

```
 ${issue.status}
```

- To access data that is stored in the parent case context data, use the keyword `parentIncidentContext`.

#### Example 45.

To access the `hostname` value in the case context data, use the following syntax:

```
 ${parentIncidentContext.hostname}
```

- Set a breakpoint in a playbook that reviews context data after a specific task.

This is available when using the debugger. As context data may be updated during a playbook run, setting a breakpoint enables you to pause the playbook execution, review the context data, and take action if necessary. Breakpoints can be useful when designing and troubleshooting playbooks. For more information, see Test your playbook.

- Add a task that writes playbook data to the case context.

When you add data to the case context, you can use this data to run playbooks on any of the issues that are included in the case.

To write playbook data to the case context, use the `setParentIncidentContext` script in a standard task. For more information, see Add context data to a case.

#### CAUTION:

Users with Trigger Playbook permissions on a given issue may still be able to modify the parent case via commands and scripts, even without full access to the case.

For more information about playbooks, see Playbooks overview.

#### Context data in sub-playbooks

By default, the context data for sub-playbooks is stored in a separate context key. Consider the following information:

- When a task in a main playbook accesses context data, it does not have direct access to sub-playbook data.
- When a task in a sub-playbook accesses context data, it does not have direct access to the main playbook data.
- If the sub-playbook has been configured to share globally, the sub-playbook context data is available to the main playbook and vice versa.

#### 3.9 | Jobs

##### Abstract

Create a time-triggered job or event-triggered job to run a playbook



Schedule playbooks to run automatically by defining a job based on events or specific times. For instance, process indicators automatically upon ingestion and then add them to your SIEM.

### 3.9.1 | Manage jobs

#### Abstract

Jobs run playbooks and are either time-triggered (run at specific times) or event triggered (run when there are changes to a feed).

A job is an automated playbook task or set of playbook tasks that are scheduled to run at predefined intervals or under specific conditions. Jobs can be used for data enrichment, periodic reporting, threat intelligence gathering, or any repetitive operational tasks that need to be performed regularly without manual intervention. There are two types of jobs:

- Time triggered jobs that run at specific times: For example, you can schedule a time triggered job that runs nightly and removes expired indicators.
- Jobs triggered by a delta or change in a feed: For example, you can define an event triggered job to run a playbook when a specified TIM feed finishes a fetch operation for new indicators.

#### NOTE:

- Only Account Admins and Instance Administrator roles can create jobs and view/edit job runs.
- If the owner of a job has been removed from Cortex AgentiX, the job will fail to run. In this case, you must change the owner to a current user.

On the Jobs page, you can:

Action	Details
Create a new job	Click + New Job.
Edit an existing job	In the table, select a job and click Edit.
Perform additional job management	In the table, select a job and click one of the following: <ul style="list-style-type: none"><li>• Run now</li><li>• Disable</li><li>• Enable</li><li>• Pause</li><li>• Resume</li><li>• Abort</li><li>• Delete</li></ul>
View job status	The chart panel at the top of the Jobs page shows various status buttons. Click one of the following buttons to filter the list of jobs for that status: <ul style="list-style-type: none"><li>• Running</li><li>• Waiting</li><li>• Error</li><li>• Disabled</li><li>• Time Triggered</li><li>• Event Triggered</li></ul> You can hide this panel by clicking Hide Chart Panel.
Search for a specific job	Enter a search query in the filter field. You can also save a filter.



Action	Details
View job details in the table	<p>By default, the displayed table columns are:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Job Status</li> <li>• Last Run</li> <li>• Next Run</li> <li>• Description</li> <li>• Playbook</li> </ul> <p>Click  to change the displayed columns. You can also select to show:</p> <ul style="list-style-type: none"> <li>• Owner</li> <li>• ID</li> <li>• Trigger</li> <li>• Job Schedule: This column shows a human readable description of a cron schedule for a job.</li> <li>• Attachments</li> </ul>

### 3.9.2 | Create a time triggered job

#### Abstract

Create a time triggered or feed triggered job in Cortex AgentiX to run a playbook.

Time triggered jobs run at predetermined times. You can schedule the job to run at a recurring time or one time at a specific date and time.

1. Select Investigation & Response → Automation → Jobs → New Job.

2. Select Time triggered.

3. If you want the job to repeat at regular intervals, select Recurring and select the desired interval.

You can choose to run the job every X number of days, on specific days of the week, at a specific time and also choose a start date and an expiration date.

You can configure the recurring job using a cron expression. To do so, after selecting the Recurring checkbox, click Switch to Cron view and enter the expression. For help defining the cron expression, click Show cron examples after switching to cron view.

#### NOTE:

To view a human readable description of a cron schedule for an existing job, click  and select Job Schedule from the available columns.

4. If you do not want the job to repeat, Select date and time for the job to run.

5. In the BASIC INFORMATION, section, add relevant time triggered job parameters from the following:

Name	Description
Name	Enter a meaningful name for the job.
Playbook	Determine which playbook to run when this job is triggered.
Description	Enter a meaningful description of the job.

6. In the QUEUE HANDLING section, select one of the following response options to use if the job is triggered while a previous run of the job is active:



- Don't trigger a new job run
- Cancel the previous job run and trigger a new job run
- Trigger a new job run and execute concurrently with the previous run

**IMPORTANT:**

We recommend to avoid triggering a job while a previous run of the job is active by configuring the playbook a job triggers to close the investigation before running a new instance of the job.

7. Select Create new job.

### 3.9.3 | Create a job triggered by a delta in a feed

#### Abstract

Create a job that is triggered when a feed has complete an operation and there is a change in the content.

Jobs triggered by a delta in a feed (event triggered jobs) run when a feed completes an operation and there is a change in the content. For the job to trigger, there must be a delta between the incoming feed and the previous one. You can define a job to trigger a playbook when the specified feed or feeds finish a fetch operation that includes a modification to the feed. The modification can be a new indicator, a modified indicator, or a removed indicator. For example, you may want to update your firewall every time a URL is added, modified, or removed from the Office 365 feed. You can configure a job that triggers the firewall update playbook to run whenever a modification is made to the feed.

**NOTE:**

A job triggered by a delta in a feed runs only if there is a change in the feed, and does not run on a feed's initial fetch. For the initial fetch, you can run the playbook manually and then set up an event triggered job for subsequent fetches.

If you want to trigger a job after a feed completes a fetch operation and the feed does not change frequently, you can select the Reset last seen option in the feed integration instance. The next time the feed fetches indicators, it will process them as new indicators in the system.

1. Select Investigation & Response → Automation → Jobs → New Job.
2. Select Triggered by delta in feed.
3. In the Trigger section, select one of the following:
  - Any feed: The playbook runs when a modification is made to any feed.
  - Specific feeds: Select the feed instances that will trigger the playbook to run when a modification is made to them.
4. In the BASIC INFORMATION section:
  - Add a meaningful name for the job.
  - Select the playbook you want to run when the conditions for the job are met.
5. Create new job.

### 3.10 | Lists

#### Abstract

Use lists to store data for use in playbooks and scripts.

Create and edit lists for use in playbooks and scripts.

A list is a data container for storing data and is mainly used in *playbooks* and *scripts* but can be accessed anywhere the context button appears (double-curly brackets). For example, in a playbook task, access the data in a list via the context button under Lists, or by using the path  `${lists.<list_name>}` . Different types of data can be stored in a list, for example, text, string, numbers, Markdown, HTML, CSS, and JSON objects.

**NOTE:**

The maximum size of a list is 209715 characters.

#### Use cases

The following are use cases for lists:



- **Defining HTML templates:** An HTML template can be defined as part of a communication task.
- **Configuring Automation Exclusion Policies:** Create lists of critical assets that should be excluded from automated remediation. For more information, see Manage automation exclusion policies.
- **Organizing Network Security:** Use lists to keep track of internal networks and their corresponding IP addresses. Compare them to a set list to ensure only authorized connections are allowed through.
- **Store Data Objects:** For example, a list of URLs, which you can call as an input for scripts and playbooks.
- **Prioritizing Case Response:** Create lists to identify critical assets, such as important users or servers. This helps improve incident management by prioritizing the most important incidents.

### 3.10.1 | Create a list

#### Abstract

Create a list that can be accessed later, such as in a playbook script or managed in the CLI (War Room or Playground).

To create a Text, Markdown, HTML, CSS, or JSON list type:

1. Go to Settings → Configurations → Object Setup → Lists → Add a List.
2. Enter a name for the list.
3. From the list, select the Content Type.
4. Set Permissions for the list. By default, all roles can read and edit the list. You can instead choose specific roles to have read only or read and edit access.
5. Add content as required. For an example of a JSON list and how to use it, see Use cases: JSON lists.
6. To save, do one of the following:
  - Click Save.
  - Click Save Version to save your changes in Version history for all Lists. This allows you to revisit and restore previous versions.

#### **NOTE:**

If you intend to use the list as part of an Automation Exclusion Policy, we recommend choosing specific roles for read and edit access. For more information on using lists in exclusion policies, see Manage automation exclusion policies.

#### **NOTE:**

If you want to edit a list from a content pack, you need to duplicate or detach a list. Detached lists do not receive updated content in subsequent Cortex AgentX content releases. To retain an updated list, reattach it.

### 3.10.2 | List commands

#### Abstract

Use list commands in the War Room, Playground, playbooks, and scripts.

Use the following list commands in the CLI in the War Room and Playground, scripts, and playbook tasks:

Command	Description	Arguments
getList	Retrieves the contents of the specified list.	listName: The name of the list for which to retrieve the contents.
createList	Creates a list with the supplied data and overwrites any existing list data.	<ul style="list-style-type: none"> <li>• listName: The name of the list to which to add items.</li> <li>• listData: The data to add to the new list and overwrites any existing list data.</li> </ul>



Command	Description	Arguments
addToList	Appends the supplied items to the specified list. If you add multiple items, make sure you use the same list separator that the list currently uses, for example, a comma or a semicolon.	<ul style="list-style-type: none"> <li>listName: The name of the list to which to append items.</li> <li>listData: The data to add to the specified list. The data will be appended to the existing data in the list.</li> </ul>
setList	Adds the supplied data to the specified list and overwrites existing list data.	<ul style="list-style-type: none"> <li>listName: The name of the list to which to add items.</li> <li>listData: The data to add to the specified list. The data overwrites the existing data in the list.</li> </ul>
removeFromList	Removes a single item from the specified list.	<ul style="list-style-type: none"> <li>listName: The name of the list from which to remove an item.</li> <li>listData: The item to remove from the specified list.</li> </ul>

Example 46.

In this example, a manageOOUsers script uses the `getList`, `createList`, and `setList` commands.

```
register_module_line('ManageOOUsers', 'start', __line__())

def _get_current_user():
    current_username = demisto.executeCommand("getUsers", {"current": True})
    if isError(current_username):
        demisto.debug(f"failed to get current username - {get_error(current_username)}")
        return
    else:
        return current_username[0]["Contents"][0]['username']

def main():
    # get current time
    now = datetime.now()

    # args
    list_name = demisto.getArg("listname")
    username = demisto.getArg("username")

    option = demisto.getArg("option")
    days_off = now + timedelta(days=int(demisto.getArg("daysoff")))
    off_until = days_off.strftime("%Y-%m-%d")

    # update list name to start with 'OOO', so we can't overwrite other lists with this
    if not list_name.startswith("OOO"):
        list_name = f"OOO {list_name}"

    current_user = _get_current_user()
    if not current_user and not username:
        return_error('Failed to get current user. Please set the username argument in the script.')

    if not username:
        # Current user was found, running script on it.
        username = current_user
    else:
        # check if provided username is a valid user
        users = demisto.executeCommand("getUsers", {})
        if isError(users):
            return_error(f'Failed to get users: {str(get_error(users))}')
        users = users[0]['Contents']

        users = [x['username'] for x in users]
        if username not in users:
            return_error(message=f"{username} is not a valid user")

    # get the out of office list, check if the list exists, if not create it:
    ooo_list = demisto.executeCommand("getList", {"listName": list_name})[0]["Contents"]
    if isError(ooo_list):
        return_error(f'Failed to get users out of office: {str(get_error(ooo_list))}')

    if "Item not found" in ooo_list:
        demisto.results(demisto.executeCommand("createList", {"listName": list_name, "listData": []}))
        ooo_list = demisto.executeCommand("getList", {"listName": list_name})[0]["Contents"]

    # check status of the list, and add/remove the user from it.
    if not ooo_list:
        list_data = []
    else:
        list_data = json.loads(ooo_list)
    if option == "add":
        # check if user is already in the list, and remove, to allow updating
```



```

list_data = [i for i in list_data if not (i['user'] == username)]
list_data.append({"user": username,
                  "offuntil": off_until,
                  "addedby": current_user if current_user else 'DBot'})
else:
    # remove the user from the list.
    list_data = [i for i in list_data if not (i['user'] == username)]

set_list_res = demisto.executeCommand("setList", {"listName": list_name, "listData": json.dumps(list_data)})
if isError(set_list_res):
    return_error(f'Failed to update the list {list_name}: {str(get_error(set_list_res))}')

# welcome back, or see ya later!
if option == "add":
    demisto.results(f"Vacation mode engaged until {off_until}, enjoy the time off {username}")
else:
    demisto.results(f"Welcome back {username}, it's like you never left!")

if __name__ in ('__builtin__', 'builtins', '__main__'):
    main()

register_module_line('ManageOOUsers', 'end', __line__())

```

### 3.10.3 | Use cases: JSON lists

#### Abstract

Manage JSON lists in Cortex AgentiX that can be accessed by automations, playbooks, etc. List commands, lists arrays separators delimiters

List data can be stored in various structures, including JSON format. When accessing a valid JSON file from within a *playbook*, it is automatically parsed as a JSON object (list). Depending on how you store the data, you may need to transform a list into an array. For example, when using non-built-in commands in a script or looping over items in a list, we recommend converting the list into an array. Working with JSON files in playbooks typically involves the following:

- Extract the data from a JSON object
- Extract a subset of the data
- Filter extracted data
- Apply transformers to extracted data.

#### Extract data from a JSON object

Create a JSON list and use the **Set** automation to create a new context key that can extract the data from the list.

##### 1. Create a List:

- In the Name field, type **Test1**.
- Select Settings → Configurations → Object Setup → Lists → Add a List.
- In the Content Type field, select JSON and add the following content:

```
{
  "domain": {
    "name": "mwidomain",
    "prod_mode": "prod",
    "user": "weblogic",
    "admin": {
      "servername": "AdminServer",
      "listenport": "8001"
    },
    "machines": [
      {
        "refname": "Machine1",
        "name": "MWINODE01"
      },
      {
        "refname": "Machine2",
        "name": "MWINODE02"
      }
    ],
    "clusters": [
      {
        "refname": "Cluster1",
        "name": "App1Cluster",
        "machine": "Box1"
      },
      {
        "refname": "Cluster1",
        "name": "App2Cluster",
        "machine": "Box2"
      }
    ],
    "servers": [

```



```

    {
      "name": "ms1",
      "port": 9001,
      "machine": "Box1",
      "clusterrefname": "Cluster1"
    },
    {
      "name": "ms2",
      "port": 9002,
      "machine": "Box2",
      "clusterrefname": "Cluster2"
    }
  ]
}

```

d. Save the list.

2. Create a playbook task with the Set automation:

- Select Investigation & Response â—> Automation â—> Playbooks â—> New Playbook.
- Name the playbook, and click Save.
- Click Create Task and provide a task name.
- In the Choose Script field, select Set .

The Set script sets a value in context under the key entered.

e. In the key field, define a context key name for the data. For example, JSONData.

Script: Set

key\* JSONData

value\*

f. In the value field, set the list you want to extract by clicking the curly brackets.

- Click Filters And Transformers.
- In the Get field, click the curly brackets, and in the Select source for value section, select the list you created in step 1: Test1.
- In the Fetch data field, select an issue to test the data.
- Click Test.

In this example, the test results have found the list data.



### Test result

```
✓ root: {} 1 item
  ✓ domain: {} 7 items
    ✓ admin: {} 2 items
      listenport: 8001
      servername: AdminServer
    ✓ clusters: [] 2 items
      ✓ 0: {} 3 items
        machine: Box1
        name: App1Cluster
        refname: Cluster1
      ✓ 1: {} 3 items
        machine: Box2
        name: App2Cluster
        refname: Cluster1
```

k. When the test completes, click Save.

l. Save the task and playbook.

3. Check all the data is stored in the context key you defined by testing the playbook using the debugger:

a. Click Run.

b. Open the Debugger Panel.

The key you defined, JSONData, holds the data in context from the JSON object.



## DEBUGGER PANEL

[Context](#) [Indicators](#)

Test data: [New Mock Alert](#)

Search in JSON context data...

[Alert](#) [Incident](#)

```
1 ~   JSONData: {  
2 ~     domain: {  
3 ~       admin: {  
4 ~         listenport: "8001"  
5 ~        servername: "AdminServer"  
6 ~       }  
7 ~     clusters: [  
8 ~       0: {  
9 ~         machine: "Box1"  
10 ~        name: "App1Cluster"  
11 ~        refname: "Cluster1"  
12 ~       }  
13 ~       1: {  
14 ~         machine: "Box2"  
15 ~        name: "App2Cluster"  
16 ~        refname: "Cluster1"
```

Extract a subset of the data

In a playbook, you can extract subsets of context data to analyze a specific information set. This approach also applies when working with lists, such as extracting a subset of data from a JSON object. In this example, we extract server information from the list created above.

1. In a playbook, create a task.
  - a. In the Choose Script field, select Set .
  - b. In the key field, define a context key name for the data; for example, `JSONDataSubset`.
  - c. In the value field, set the list you want to extract by clicking the curly brackets.
  - d. Click Filters And Transformers.
  - e. In the Get field, enter `lists.Test1.domain.servers`.
  - f. In the Fetch data field, select an issue to test the data.
  - g. Click Test.
  - h. When the test completes, click Save.
  - i. Save the task and the playbook.
2. Check that all the data is stored in the context key you defined by testing the playbook using the debugger.



- Click Run Debugger Panel.
- The key you defined (JSONDataSubset) holds the subset of the data in context from the JSON object.

The screenshot shows the 'Context' tab selected in the JSON Context tool. The 'Test data' dropdown is set to 'New mock incident'. A search bar at the top contains the placeholder 'Search in JSON context data...'. Below the search bar are 'Collapse' and 'Expand' buttons. The extracted data is displayed as follows:

```

v JSONDataSubset:
  v 0:
    clusterrefname: Cluster1
    machine: Box1
    name: ms1
    port: 9001
  v 1:
    clusterrefname: Cluster2
    machine: Box2
    name: ms2
    port: 9002
  
```

#### Filter extracted data

You can filter the extracted data subset to analyze it on a more granular level. In this example, we filter Box1 information from the list created in Extract the data from a JSON Object above.

- Re-open the task you created above.
- Click the value field.
- Under Filter, click Add Filter.
- Set the condition you want to filter.

In this example, retrieve the list of machines named `Box1` from `Test1` list by setting the filter `lists.Test1.domain.servers.machine Equals Box1`.

The screenshot shows the 'FILTERS & TRANSFORMERS FOR value' dialog. It has a header with a close button (X). The main area is divided into three sections:

- 1 Get**: Shows the path `lists.Test1.domain.servers` with a copy icon (C) to its right.
- 2 Filter**: (Get subset of the data, e.g. `File.Type is PDF`)  
Where all of the following are true for `lists.Test1.domain.servers`  
Condition: `lists.Test1.domain.servers.machine Equals Box1` (with a delete icon)
- 3 Apply transformers on the field (Optional)**: Contains a '+ Add transformer' button.

- Click Test.
- Check whether the data subset was accessed successfully by selecting the data source from an issue. You can see the results returned `machine: Box1`.



The screenshot shows the 'TEST' interface. On the left, a sidebar lists categories: DBotScore (6 items), File (3 items), IP (5 items), IPInfo (1 item), VirusTotal (2 items), and alert (107 items). Below this is a 'Test' button. The main area displays a 'Test result' pane with a tree view. The root node has two items, one of which is expanded to show four items. Underneath are clusterrefname: Cluster1, machine: Box1, name: ms1, and port: 9001. At the bottom right of the result pane is a 'Done testing' button.

#### Apply transformers to extracted data

In general, in a playbook task, you can transform (apply changes) to the data extracted. This also applies to working with lists ,such as transforming extracted data from a JSON object. In this example, we extract the first element from the list created in the 'Extract Data from a JSON Object' section above and transform it to uppercase.

1. Re-open the task, click the contents of the value field, and keep the current filters.
2. In the Apply transformers on the field, click Add transformer.
3. Add the following transformers to the extracted data:

1. Add the `Get index (General)` transformer to extract a specific machine element.

Set `index: 0` to extract the first element from the list.

2. Add the `To upper case (String)` transformer.

The `To upper case (String)` transformer does not work on lists, only on individual elements. Therefore, the `Get index (General)` transformer should be applied before adding the `To upper case (String)` transformer.

#### 1 Get

`lists.Test1.domain.servers.machine`

#### 2 Filter (Get subset of the data, e.g.

`lists.Test1.domain.servers`

File.Type is PDF

Where all of the following are true for  
`lists.Test1.domain.servers`

`lists.Test1.domain.servers.machine Equals Box1`

+ Add filter

#### 3 Apply transformers on the field (Optional)

`Get index (index: 0)`

`To upper case`

4. In the Fetch Data field, select an issue to test and click Test.

#### 3.10.4 | Transform a list into an array

##### Abstract

Create a transformer to split a list into an array when adding or editing a task in a playbook or when mapping an integration instance in Cortex AgentiX.

Create a transformer to split a list into an array, add or edit a task in a *playbook*, or map an instance.

1. Go to Investigation & Response â Automation â Playbooks and create or edit a playbook.



2. Select Create Task.
  3. In the Choose script field, select the Set automation.
  4. In the Key field, enter the key name.
  5. In the value field, click {}
  6. Add a transformer.
    - a. Click Filters And Transformers.
    - b. In the Get field, click {}.
    - c. Expand the Lists node and select a list to transform.
    - d. In Apply transformers on the field, click Add transformer.
    - e. Search for and select Split.
    - f. (Optional) In the delimiter field, type the delimiter used to separate the items in the string (default is ",").
  - g. Click Save.
7. Save the task and playbook.

## 3.11 | Data management

### Abstract

Learn how to use datasets, data model rules, and configure parsing rules.

Use datasets, data models, and parsing rules in Cortex AgentiX.

### 3.11.1 | Dataset management

#### Abstract

Learn more about managing your datasets and understanding your overall data storage, period-based retention.

#### PREREQUISITE:

Dataset Management requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Parsing Rules, Data Model Rules, and Event Forwarding.

The Dataset Management page enables you to manage your datasets and understand your overall data storage duration for different retention periods and datasets based on your hot storage license, and retention add-ons that extend your storage. You can view details about your Cortex AgentiX licenses and retention add-ons by selecting Settings → Cortex AgentiX License. For more information on license retention and the defaults provided per license, see Data retention policy.

#### IMPORTANT:

Cortex AgentiX enforces retention on all log-type datasets, excluding metrics and users.

#### Hot storage

Your current hot storage license, including the default license retention and any additional retention add-ons to extend storage, are listed within the Hot Storage License section of the Dataset Management page. Whenever you extend your license retention, depending on your requirements and license add-ons for hot storage, the add-ons are listed.

#### NOTE:

Cold storage, in addition to a cold storage license, requires compute units (CU) to run cold storage queries. For more information on CU, see Manage compute units.

#### Additional hot storage

You can expand your license retention to include flexible Hot Storage based retention to help accommodate varying storage requirements for different retention periods and datasets. This add-on license is available to purchase based on your storage requirements for a minimum of 1,000 GB. If this license is purchased, an Additional Storage subheading in the Hot Storage License section is displayed on the Dataset Management page with a bar indicating how much of the storage is used.

#### NOTE:

Only datasets that are already handled as part of the GB license are supported for this license. In addition, the retention configuration is only available in Cortex AgentiX, as opposed to the public APIs.

#### Edit the retention plan



On any dataset configured to use Additional Hot Storage, you can edit the retention period. This enables you to view the current retention details for hot and cold storage and configure the retention. This includes setting the amount of flexible hot storage-based retention designated for a dataset and the priority for the dataset's hot storage.

#### How to edit the retention plan

1. Select Settings → Configurations → Data Management → Dataset Management.
2. In the Datasets table, right-click any dataset designated with flexible hot storage, and select Edit Retention Plan.
3. Set the following parameters:
  - Additional hot storage: Set the amount of flexible hot storage-based retention designated for this dataset in months, where a month is calculated as 31 days.
  - Hot Storage Priority: Select the priority designated for this dataset's hot storage as either Low, Medium, or High.
4. Click Save.

#### Datasets table

For each dataset listed in the table, the following information is available:

##### **NOTE:**

- Certain fields are exposed and hidden by default. An asterisk (\*) is beside every field that is exposed by default.
- Datasets include dataset permission enforcements in the Cortex Query Language(XQL), Query Center, and XQL Widgets. For example, to view or access any of the **endpoints** and **host\_inventory** datasets, you need role-based access control (RBAC) permissions to the Endpoint Administration and Host Inventory views. Managed Security Services Providers (MSSP) administration permissions are not enforced on child tenants, but only on the MSSP tenant.

Field	Description
*TYPE	Displays the type of dataset based on the method used to upload the data. The possible values include: Correlation, Lookup, Raw, Snapshot, System, and User. For more information on each dataset type, see <a href="#">What are datasets?</a> .
*LOG UPDATE TYPE	Event logs are updated either continuously (Logs) or the current state is updated periodically (State) as detailed in the Last Updated column.
*LAST UPDATED	Last time the data in the dataset logs were updated.  <b>IMPORTANT:</b> This column is updated once a day. Therefore, if the dataset was created or updated by the target or lookup flows, it's possible that the Last Updated value is a day behind when the queries or reports were run as it was before this column was updated.
*ADDITIONAL STORAGE	Amount of flexible hot storage-based retention designated for this dataset in months, where a month is calculated as 31 days.
*TOTAL DAYS STORED	Actual number of days that the data is stored in the Cortex AgentiX tenant, which is comprised of the HOT RANGE.
*HOT RANGE	Details the exact period of the Hot Storage from the start date to the end date.
*TOTAL SIZE STORED	Actual size of the data that is stored in the Cortex AgentiX tenant. This number is dependent on the events stored in the hot storage. For the <b>xdr_data</b> dataset, where the first 31 days of storage are included with your license, the first 31 days are not included in the TOTAL SIZE STORED number.
*ADDITIONAL SIZE STORED	Actual size of the additional flexible hot storage data that is stored in the Cortex AgentiX tenant in GB. This number is dependent on the events stored in the hot storage.



Field	Description
*AVERAGE DAILY SIZE	Average daily amount stored in the Cortex AgentiX tenant. This number is dependent on the events stored in the hot storage.
*HOT STORAGE PRIORITY	Indicates the priority set for the dataset's hot storage as either Low, Medium, or High.
*TOTAL EVENTS	Number of total events/logs that are stored in the Cortex AgentiX tenant. This number is dependent on the events stored in the hot storage.
*AVERAGE EVENT SIZE	Average size of a single event in the dataset (TOTAL SIZE STORED divided by the TOTAL EVENTS). This number is dependent on the events stored in the hot storage.
*TTL	<p>For lookup datasets, displays the value of the time to live (TTL) configured for when lookup entries expire and are removed automatically from the dataset. The possible values are:</p> <ul style="list-style-type: none"> <li>• Forever: Lookup entries never expire (default).</li> <li>• Custom: Lookup entries expire according to a set number of days, hours, and minutes. The maximum number of days is 99999.</li> </ul> <p>For more information, see Set time to live for lookup datasets.</p>
DEFAULT QUERY TARGET	Details whether the dataset is configured to use as your default query target in XQL Search, so when you write your queries you do not need to define a dataset. By default, only the <code>xdr_data</code> dataset is configured as the DEFAULT QUERY TARGET and this field is set to Yes. All other datasets have this field set to No. When setting multiple default datasets, your query does not need to mention any of the dataset names, and Cortex AgentiX queries the default datasets using a <code>join</code> .
TOTAL HOT RETENTION	Total hot storage retention configured for the dataset in months, where a month is calculated as 31 days.

#### Dataset views

Cortex AgentiX supports creating dataset views in the [Dataset Management](#) page to enhance data efficiency and security. Dataset views provide a virtual representation of data from one or more datasets, based on the Cortex Query Language (XQL) query defined, and provide multiple benefits, such as joining datasets into logical subsets through defined queries, manipulating data without altering underlying datasets, and segregating data for specific user needs or access privileges through the Role-based access control (RBAC) settings.

Once a dataset view is created, you can edit or delete the dataset view by right-clicking the dataset view in the Dataset Views table. A dataset view can only be deleted if there are no other dependencies. For example, if a Correlation Rule is based on a dataset view, you wouldn't be able to delete the dataset view until you removed the dataset view from the XQL query of the Correlation Rule.

Cortex AgentiX logs entries for events related to creating, editing, and deleting datasets or dataset views. These monitored activities are available to view in the datasets and dataset views audit logs in the Management Audit Logs. For more information, see [Monitor datasets and dataset views activity](#).

#### Building XQL dataset view queries

When building an XQL query to define a dataset view, the query is built in the same way as creating a query through the Query Builder. Yet, it's important to be aware of the following points that are specific for dataset view queries:



- The following features are unsupported in dataset view queries:
  - RT Correlation Rules
  - Cortex Data Model (XDM)
  - Query Library
  - Presets
- Only the following XQL stages are supported when building a dataset view query:
  - alter
  - dedup
  - fields
  - filter
  - join
  - replacenull
  - union
- Once the dataset view is created, it is listed as an available **dataset** when building your XQL queries as long as you have the necessary permissions to access the dataset view in the Role-based access control (RBAC) settings.

How to create a dataset view

1. Select Settings → Configurations → Data Management → Dataset Management → Dataset Views.
2. Click New Dataset View.
3. Enter a Name and Description (optional) for the dataset view.
4. Create your XQL query for the dataset view by typing in the query box.
5. (Optional) Click Run to view the query results.

The query must contain no errors, including using only supported commands, to run; otherwise, the Run button remains disabled.

6. Click Save.

**NOTE:**

You'll only be able to save the dataset view if the query contains no errors; otherwise, the Save button is disabled.

Once the dataset view is created, you can now control user access permissions through Role-based access control (RBAC).

Dataset views access permissions

**LICENSE TYPE:**

Managing Roles requires an Account Admin or Instance Administrator role.

Access permissions for dataset views are configured in the same way that you set dataset access permissions for any dataset through user roles in Cortex AgentiX Access Management. Cortex AgentiX uses role-based access control (RBAC) to manage roles with specific permissions for controlling user access. RBAC helps manage access to Cortex AgentiX components and datasets, so that users, based on their roles, are granted minimal access required to accomplish their tasks. Once the user role is configured to access these dataset views, you can now assign the user role to the designated users or user groups, who want to access these dataset views.

How to set access permissions for dataset views

1. Select Settings → Configurations → Access Management.
2. Configure a user role with the dataset views that you want users to access.
  - a. Select Roles.
  - b. You can perform one of the following:
    - To create a new role to assign the dataset views, click New Role, and set a Role Name and Description (optional).
    - To edit an existing user role with these dataset views, right-click the relevant user role, and select Edit Role.
    - To create a new role based on an existing role, right-click the relevant user role, select Save As New Role, and set a Role Name and Description (optional).
  - c. Under Datasets, you have two options for setting the Cortex Query Language (XQL) dataset access permissions for the user role:



- Set the user role with access to all XQL datasets by disabling the Enable dataset access management toggle.
  - Set the user role with limited access to certain XQL datasets by selecting the Enable dataset access management toggle and selecting the datasets under the different dataset category headings.
- d. Scroll down to Dataset View and select the particular dataset views that you want assigned to this user role.
- e. Click Save.

3. Assign the user role with the dataset views configured to the designated users or user groups. For more information, see Create a Role in Manage users in the Cortex AgentiX tenant.

#### Dataset Views table

For each dataset view listed in the table, information is available. Here are descriptions on the columns that may require further explanation:

Field	Description
SOURCE QUERY	Displays the query used to create the dataset view.
IS VALID	Details whether the query for the dataset view is still valid or not.
RELATED TABLES	Details the other datasets that are related to this dataset view.

#### 3.11.1.1 | What are datasets?

##### Abstract

Learn how to import, delete, and interact with custom or third-party datasets in Cortex AgentiX.

##### PREREQUISITE:

Dataset Management requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Parsing Rules, Data Model Rules, and Event Forwarding.

Cortex AgentiX runs every Cortex Query Language (XQL) query against a dataset. A dataset is a collection of column:value sets. If you do not specify a dataset in your query, Cortex AgentiX runs the query against the default datasets configured, which is by default `xdr_data` for a dataset query. The `xdr_data` dataset contains all of the endpoint and network data that Cortex AgentiX collects. For a Cortex Data Model (XDM) query, unless specific datasets are specified, a query will run against all mapped datasets. You can always change the default datasets using the set to default option. You can also upload datasets as a CSV, TSV, or JSON file that contains the data you are interested in querying. These uploaded datasets are called lookup datasets.

It's also possible to create dataset views, which provide a virtual representation of data from one or more datasets, based on the Cortex Query Language (XQL) query defined. Dataset views enhance data efficiency and security. For example, by segregating data for specific user needs or access privileges through the Role-based access control (RBAC) settings. For more information, see Dataset views.

To query other datasets, you have the following options:

- Set a dataset as default, which enables you to query the datasets without specifying them in the query.
- Name a specific dataset at the beginning of your query with the `dataset` stage command.

##### Dataset types

The type of dataset is based on the method used to upload the data. The possible types include:

- Correlation: A dataset containing data saved from a correlation rule.
- Lookup: A dataset containing key-value pairs that can be used as a reference to correlate to events. For example, a user list with corresponding access privileges. You can import or create a lookup dataset, and then reference the values for a certain key, run queries and take action. For more information, see Lookup datasets.
- Raw: Every dataset where PANW data is ingested out-of-the-box or third-party data is ingested using a configured dedicated collector.
- Snapshot: A dataset that contains only the last successful snapshot of the data, such as Workday or ServiceNow CMDB tables.
- System: Cortex AgentiX datasets that are created out-of-the-box.
- User: If saved by a query using the `target` command, the Type can be either User or Lookup.

##### Datasets in XQL

##### IMPORTANT:



**By default, forensic datasets are not included in XQL query results, unless the dataset query is explicitly defined to use a forensic dataset.**

Cortex Query Language (XQL) supports using different languages for dataset and field names. In addition, when setting up your XQL query, it is important to keep in mind the following:

- The dataset formats supported are dependent on the data retention offerings available in Cortex AgentiX according to whether you want to query hot storage or cold storage.
- Dataset refresh times: While most out-of-the-box system datasets are ingested in near real-time, the following datasets have specific refresh schedules.
  - **endpoints**: Refreshed every hour.
  - **pan\_dss\_raw**: Refreshed daily.
  - Forensics datasets: Data collection behavior depends on your Agent Settings profile.
    - Default: Data is collected as a one-time snapshot and does not update.
    - Scheduled: If you specify a collection interval, the value represents the number of hours between updates, such as an interval of 24 equals once per day.
    - Minimum: The shortest allowable interval is 12 hours.
- Query against a dataset by selecting it with the **dataset** command when you create an XQL query. For more information, see Create XQL query.
- After your query runs, you can always save your query results as a dataset. You can use the target stage command to save query results as a dataset.
- Schema changes to datasets may not be reflected in the autocomplete suggestions and definitions as you type in real time the XQL query and can appear with a slight delay.

#### Managing datasets and dataset views

You can manage your datasets and dataset views in Cortex AgentiX from the Settings → Configurations → Data Management → Dataset Management page.

Below are some of the main tasks available for all dataset types by right-clicking a particular dataset or dataset view listed in either the Datasets or Dataset Views table. Only tasks that need further explanation are explained below. Datasets and dataset views can only be deleted if there are no other dependencies. For example, if a Correlation Rule is based on a dataset or dataset view or dataset view, you wouldn't be able to delete the dataset or dataset view until you removed the dataset view from the XQL query of the Correlation Rule.

#### **NOTE:**

For more information on tasks specific to lookup datasets, see [Lookup datasets](#).

#### View Schema

Select View Schema to view the schema information for every field found in the dataset or dataset view result set in the Schema tab after running the query in XQL. Each system field in the schema is written with an underscore (\_) before the name of the field in the FIELD NAME column in the table.

#### **NOTE:**

Schema changes to datasets may not be reflected in the autocomplete suggestions and definitions as you type in real time the XQL query and can appear with a slight delay.

#### Set as default

Select Set as default to query the dataset without having to specify it in your queries in XQL by typing **dataset = <name of dataset>**. Once configured, the DEFAULT QUERY TARGET column entry for this dataset is set to Yes in the Datasets table. By default, this option is not available when right-clicking the **xdr\_data** dataset as this dataset is the only dataset configured as the DEFAULT QUERY TARGET as it contains all of the endpoint and network data that Cortex AgentiX collects. Once you Set as default another dataset, you can always remove it by right-clicking the dataset and selecting Remove from defaults. When setting multiple default datasets, your query does not need to mention any of the dataset names, and Cortex AgentiX queries the default datasets using a **join**. This option is only relevant for datasets.

#### Copy text to clipboard

Select Copy text to clipboard to copy the name of the dataset or dataset view to your clipboard.

#### 3.11.1.2 | [Lookup datasets](#)

##### Abstract

Learn more about lookup datasets to correlate data from a data source with events in your environment.

#### **PREREQUISITE:**

Dataset Management requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Parsing Rules, Data Model Rules, and Event Forwarding.



Lookup datasets enable you to correlate data from a data source you provide with the events in your environment. For example, you can create a lookup with a list of high-value assets, terminated employees, or service accounts in your environment. Use lookups in your search, detection rules, threat hunting, and response playbooks. Lookups are stored as name-value pairs and are cached for optimal query performance and low latency.

Lookup tables support low-frequency changes of up to 1200 modifications per day. Changes are implemented whenever a lookup dataset is edited, where only one person or user can edit the file at a given time. Concurrent users editing the file are not supported.

#### Use case scenarios

- Investigate threats and respond to cases quickly with the rapid import of IP addresses, file hashes, and other data from CSV files. After you import the data, use lookup name-value pairs for joins and filters in threat hunting and general queries.
- Import business data as a lookup. For example, import user lists with privileged system access, or terminated employees. Then, use the lookup to create allow lists and blocklists to detect or prevent those users from logging in to the network.
- Create allow lists to suppress issues from a group of users, such as users from authorized IP addresses that perform tasks that would normally trigger the issue. Prevent benign events from becoming issues.
- Enrich event data. Use lookups to enrich your event data with name-value combinations derived from external data sources.

#### How are lookup datasets created?

You can import or create a lookup dataset, and then reference the values for a certain key, run queries, and take action. Lookup datasets are created by any of the following methods:

- Manual upload from a CSV, TSV, or JSON file to Cortex AgentiX from the Dataset Management page. For more information, see Import a lookup dataset.
- Automatic upload by the Files and Folders Collector.
- Query results are saved to a lookup dataset. If saved using the `target` stage, the Type can be either User or Lookup. For more information, see the target stage.

#### IMPORTANT:

When you create or add data to a lookup dataset using the `target` stage, the `_time` field won't be included by default unless you explicitly add it with the `fields` stage.

After a lookup, a dataset is imported, you can always edit the dataset to update the data manually by right-clicking the dataset and selecting Edit.

#### NOTE:

A lookup dataset can only be deleted if there are no other dependencies. For example, if a Correlation Rule is based on a lookup dataset, you wouldn't be able to delete the lookup dataset until you removed the dataset from the XQL query of the Correlation Rule.

#### 3.11.1.2.1 | Import a lookup dataset

##### Abstract

Learn more about importing data from an external file to create or update a lookup dataset in Cortex AgentiX.

#### PREREQUISITE:

Dataset Management requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Parsing Rules, Data Model Rules, and Event Forwarding.

You can import data from CSV, TSV, or JSON files into Cortex AgentiX to create or update lookup datasets.

#### PREREQUISITE:

When uploading a CSV, TSV, or JSON file, ensure that the file meets the following requirements:

- The maximum size for the total data to be imported into a lookup dataset is 30 MB from the Dataset Management page. Otherwise, the limit is 50 MB using Cortex Query Language (XQL) or APIs.
- Field names can contain characters from different languages, special characters, numbers (0–9), and underscores (\_).
- Field names can't exceed 128 characters.
- Field names can't contain duplicate names, white spaces, or carriage returns.
- The file doesn't contain a byte array (binary data) as it can't be uploaded.
- Each line in the JSON file must represent one JSON object. Ensure no brackets enclose the objects at the top-level.

#### Example 47.

Here's an example of a JSON file in the correct format for upload:

```
{"firstName": "NAME_1", "SurName": "NAME_11", "employeeID": {"id": "ID_AAAAA_2"}},  
{"firstName": "NAME_2", "SurName": "NAME_22", "employeeID": {"id": "ID_AAAAA_3"}},  
{"firstName": "NAME_3", "SurName": "NAME_32", "employeeID": {"id": "ID_AAAAA_4"}}
```



1. Select Settings → Configurations → Data Management → Dataset Management → + Lookup.

2. Browse to your CSV, TSV, or JSON file. You can only upload a TSV file if it contains a `.tsv` file extension.

3. (Optional) Under Name, type a new name for the target dataset.

By default, Cortex AgentX uses the name of the original file as the dataset name. You can change this name to something that will be more meaningful for your users when they query the dataset. For example, if the original file name is `mrkdptusrsnov23.json`, you can save the dataset as `marketing_dept_users_Nov_2023`.

Dataset names can contain special characters from different languages, numbers (0–9) and underscores (\_). You can create dataset names using uppercase characters, but in queries, dataset names are always treated as if they are lowercase.

#### **IMPORTANT:**

The name of a dataset created from a TSV file must always include the extension. For example, if the original file name is `mrkdptusrsnov23.tsv`, you can save the dataset with the name `marketing_dept_users_Nov_2023.tsv`.

4. Replace the existing data in the dataset overwrites the data in an existing lookup dataset with the contents of the new file.

5. Click Add to add the file as a lookup.

6. After receiving a notification reporting that the upload succeeded, Refresh  to view it in your list of datasets.

If the upload fails for any reason, you'll receive a notification in the Notification Center.

#### 3.11.1.2.2 | Download JSON file of lookup dataset

##### Abstract

Learn more about downloading a lookup dataset as a JSON file.

#### **PREREQUISITE:**

Dataset Management requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Parsing Rules, Data Model Rules, and Event Forwarding.

You can only download a JSON file for a lookup dataset, where the Type set to Lookup on the Dataset Management page. This option is not available for any other dataset type.

When you download a lookup dataset with field names in a foreign language, the downloaded JSON file displays the fields as `COL_<randomstring>` as opposed to returning the fields in the foreign language as expected.

1. Open the Settings → Configurations → Data Management → Dataset Management page.

2. In the Datasets table, right-click the lookup dataset that you want to download as a JSON file, and select Download.

#### 3.11.1.2.3 | Set time to live for lookup datasets

##### Abstract

Learn more about setting the time to live (TTL) for lookup datasets in Cortex AgentX.

#### **PREREQUISITE:**

Dataset Management requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Parsing Rules, Data Model Rules, and Event Forwarding.

You can specify when lookup entries expire and are removed automatically from the lookup dataset by configuring the time to live (TTL). The time period of the TTL interval is based on when the data was last updated. The default is forever and the entries never expire. You can also configure a specific time according to the days, hours, and minutes. Expired elements are removed from the lookup dataset by a scheduled job that runs every five minutes.

1. Open the Settings → Configurations → Data Management → Dataset Management page.

2. In the Datasets table, right-click the lookup dataset, and select Set TTL.

3. Select one of the following to configure when lookup dataset entries expire and are removed:

- Forever: Lookup entries never expire (default).
- Custom: Lookup entries expire according to a set number of days, hours, and minutes. The maximum number of days is 99999.

4. Click Save.

The TTL column in the Datasets table is updated with the changes and these changes are applied immediately on all existing lookup entries.



### 3.11.1.3 | Monitor datasets and dataset views activity

#### Abstract

Learn more about the monitored Cortex AgentiX datasets and dataset views activities.

#### PREREQUISITE:

Dataset Management requires View/Edit RBAC permissions for Data Management (under Configurations à Data Management), which are the same permissions required for Parsing Rules, Data Model Rules, and Event Forwarding.

Cortex AgentiX logs entries for events related to datasets and dataset views monitored activities. Cortex AgentiX stores the logs for 365 days. To view the datasets and dataset views audit logs, select Settings à Management Audit Logs.

You can customize your view of the logs by adding or removing filters to the Management Audit Logs table. You can also filter the page result to narrow down your search. The following table describes the default and optional fields that you can view in the Cortex AgentiX Management Audit Logs table:

#### NOTE:

Certain fields are exposed and hidden by default. An asterisk (\*) is beside every field that is exposed by default.

Field	Description
Description*	Log message that describes the action.
Email	Email of the user who performed the action.
Host Name*	This field is not applicable for datasets and dataset views logs.
ID	Unique ID of the action.
Reason	This field is not applicable for datasets and dataset views logs.
Result*	The result of the action ( Success, Fail, or N/A)
Severity*	Severity associated with the log: <ul style="list-style-type: none"><li>• Critical</li><li>• High</li><li>• Medium</li><li>• Low</li><li>• Informational</li></ul>
Timestamp*	Date and time when the action occurred.



Field	Description
Type* and Sub-Type*	<p>Additional classifications of dataset and dataset view logs (Type and Sub-Type):</p> <ul style="list-style-type: none"> <li>• Datasets: <ul style="list-style-type: none"> <li>◦ Create Dataset</li> <li>◦ Delete Dataset</li> <li>◦ Update Dataset</li> </ul> </li> <li>• Dataset Views: <ul style="list-style-type: none"> <li>◦ Create Dataset View</li> <li>◦ Delete Dataset View</li> <li>◦ Update Dataset View</li> </ul> </li> </ul>
User Name*	Name of the user who performed the action.

### 3.11.2 | What are Data Model Rules?

Abstract

Learn more about what Cortex Data Model (XDM) Rules enable you to do.

#### PREREQUISITE:

Data Model Rules requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Dataset Management, Parsing Rules, and Event Forwarding.

Cortex AgentIX enables you to map your logs into a single, unified data model. This data model provides a consolidated schema, and a simpler way to interact with your data, regardless of its source or dataset. To familiarize yourself with the data model schema, see Cortex XSIAM Data Model Schema.

You can map your data to the data model using Data Model Rules, either by using the Default Rules that are automatically added when installing Content Packages from the Marketplace, or by creating user-defined rules. You create rules with the Data Model Rules editor, which enables you to do the following:

- Map 3<sup>rd</sup> party data to a consolidated schema with predefined data types.
- Enjoy auto-complete and mapping suggestions.
- Map multiple queriable datasets to the data model.

Data Model Rules contain the following built-in characteristics:

- Each Data Model Rule is mapped between one dataset and the data model.
- A Data Model Rule takes rows from a dataset to use as an input, performs an arbitrary number of transitions and modifications on each column in the dataset using Cortex Query Language (XQL), and then returns the normalized rows with the corresponding data model's schema.

#### 3.11.2.1 | Data Model Rules editor views

Abstract

Learn about the Data Model Rules editor User Defined Rules, Default Rules, and Both views.

#### PREREQUISITE:

Data Model Rules requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Dataset Management, Parsing Rules, and Event Forwarding.

The Data Model Rules editor contains the following views.

- User Defined Rules (default): Displays an editor for writing your own custom Data Model Rules that override the default rules. Once you edit a default Data Model mapping, you will no longer receive Marketplace updates.
- Default Rules (read-only): Displays the data model rules that are provided by default.
- Both: Side-by-side view of both the Default Rules and User Defined Rules, so that you can easily view both sets of rules on the same screen.



### 3.11.2.2 | Data Model Rules file structure and syntax

#### Abstract

Learn about the Data Model Rules file structure and syntax.

#### PREREQUISITE:

Data Model Rules requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Dataset Management, Parsing Rules, and Event Forwarding.

#### File structure

The Data Model Rules file consists of multiple sections of the following two types, which also represent the custom syntax specific to Data Model Rules:

- MODEL: This section is used to define the mapping between a single dataset and the data model.
- (OPTIONAL) RULE: Rules are part of the Cortex Query Language (XQL) syntax, which are tagged with a name, and can be reused in the code in the MODEL sections, or in other RULE sections (recursively), by using [rule:ruleName].

The order of the sections is not significant.

#### Syntax

The syntax used in the Data Model Rules file is derived from XQL, with a few modifications. This subset of XQL is called *XQL for Data Modeling (XQLm)*.

#### NOTE:

For more information on XQL syntax, see the [XQL Language Reference Guide](#).

In the MODEL and RULE sections, the following modifications apply to the XQLm syntax:

- Only the following XQL stages are permitted: alter and . An additional call stage is supported, which is used to invoke another rule.

#### NOTE:

You cannot call a RULE section that exists in Default Rules from the User Defined Rules section.

- No output stages are supported.
- XDM\_ALIAS cannot be used in rules. It is only supported in queries. For more information, see the search stage.
- Every model definition in the Data Model Rules file must end with a semicolon (;).
- Each XDM field used in the MODEL and RULE sections is constructed using dot notation using the following format:

`xdm.[<context>].[<compound>].<field>`

For more information, see Field structure.

### 3.11.2.3 | MODEL

#### Abstract

Learn how to write a MODEL section in a Data Model Rules file, and about the syntax to use in the file.

A MODEL section is used to define the mapping between a single dataset and the data model. The MODEL section is mandatory per dataset. A RULE section is optional, and is used to help organize the MODEL sections.

MODEL syntax is derived from Cortex Query Language (XQL), with a few modifications, as explained in Data Model Rules file structure and syntax. In addition, MODEL sections contain the following syntax add-ons:

- You can have multiple MODEL sections.
- MODEL sections take parameters, and not names as RULE sections use, where some are mandatory and others are optional.

```
[MODEL: dataset=<dataset>, content_id=<content_id>]  
<build the XQL logic>;
```

The parameter descriptions are explained in the following table:

Parameter	Description
dataset	The name of the dataset that contains the source data to apply the mapping on (mandatory).



Parameter	Description
content_id	Identifier of the content as defined in the content package from the Marketplace. This parameter is relevant only for Default Rules and is not available in User Defined Rules (optional).

Example 48.

```
[MODEL: dataset=panw_ngfw_traffic]
filter appid = "dns"
| alter dns_helper = json_extract(event, "$.dns")
| alter xdm.network.dns.opcode = to_integer(json_extract_scalar(dns_helper, "$.opcode"),
    xdm.network.dns.is_truncated = to_boolean(json_extract_scalar(dns_helper, "$.is_truncated"))
);
```

Points to keep in mind when writing MODEL sections

- MODEL parameter names are not case-sensitive.
- Cortex Data Model (XDM) System fields (\_time, \_insert\_time, \_vendor, \_product) are mapped automatically from the dataset from the fields with the same names.
- As section order is not significant, you do not have to declare a RULE before using it in a MODEL section.
- Each field used in the MODEL and RULE sections is constructed using dot notation with a specific format. Each field must be part of the predefined field set of the data model's schema. However, temporary variables, which will not affect the modeling, may be used. For more information, see Field structure.
- A MODEL section can invoke a rule using the call stage.

Example 49.

In this example, both the RULE and MODEL sections are provided, so you can see how the call stage invokes the rule.

```
[RULE: common_ngfw_modeling]
alter xdm.source.ipv4 = json_extract_scalar(actor, "$.client_ip")
| alter xdm.network.ip_protocol = if(
    proto = 6, XDM_CONST.IP_PROTOCOL_TCP,
    proto = 11, XDM_CONST.IP_PROTOCOL_UDP,
    proto
);

[MODEL: dataset=panw_ngfw_traffic]
filter appid = "dns"
| call common_ngfw_modeling
| alter dns_helper = json_extract(event, "$.dns")
| alter xdm.network.dns.opcode = to_integer(json_extract_scalar(dns_helper, "$.opcode"),
    xdm.network.dns.is_truncated = to_boolean(json_extract_scalar(dns_helper, "$.is_truncated"))
);
```

- You can use the config case\_sensitive stage in the MODEL section to configure whether field values in the XDM are evaluated as case sensitive or case insensitive. The config case\_sensitive stage must be added at the beginning of the query. If you do not provide this stage in your query, the default behavior is false; case is not considered when evaluating field values.

#### NOTE:

The Settings → Configurations → XQL Configuration → Case Sensitivity (case\_sensitive) setting can overwrite this case\_sensitive configuration for all fields in the application except for BIOCs, which will remain case insensitive no matter what this setting is set to. For more information on this setting, see XQL Configuration.

- Cortex AgentiX enables analytics to run on the following data:

- All mapped network data to the network 5 tuple (source IP, source port, target IP, target port, IP protocol), automatically creating network stories for XDM network data.
- All mapped authentication data, automatically creating authentication stories for XDM identity data when certain mandatory fields are mapped. For more information, see How to map authentication story events?.

#### NOTE:

We recommend that you do not configure the same data source in both Marketplace and using a Cortex AgentiX data collector. Yet, if you do, the following will happen:

- For network data, all relevant logs from the different data sources are stitched to the same network story.
- For authentication data, all relevant logs from the different data sources are stitched to the same authentication story as long as the logs contain the network 5 tuple (source IP, source port, target IP, target port, IP protocol). The rest of the logs, without the network 5 tuple, create duplicate authentication stories.



### 3.11.2.4 | RULE

#### Abstract

Learn how to write a RULE section in a Data Model Rules file and about the syntax to use in the file.

Rules are very similar to functions in modern programming languages. They are essentially named pieces of Cortex Query Language (XQL) syntax, and can be reused in the code in the MODEL sections, or in other RULE sections (recursively), by using [rule:ruleName]. A RULE is an optional data model syntax.

RULE syntax is derived from XQL with a few modifications, as explained in the Data Model Rules file structure and syntax.

#### NOTE:

For more information on the XQL syntax, see the [XQL Language Reference Guide](#).

Points to keep in mind when writing RULE sections

- Rules are defined by [rule:ruleName] as shown in the following example:

Example 50.

```
[RULE: common_ngfw_modeling]
alter xdm.source.ipv4 = json_extract_scalar(actor, "$.client_ip")
| alter xdm.network.ip_protocol = if(
    proto = 6, XDM_CONST.IP_PROTOCOL_TCP,
    proto = 11, XDM_CONST.IP_PROTOCOL_UDP,
    proto
);
;
```

- Rules are invoked by using a call stage.
- Rule names are not case-sensitive.
- Rule names must be unique across the entire file.
- As section order is not significant, you do not have to declare a rule before using it. You can have the rule definition section written below other sections that use that specific rule.
- Each field used in the MODEL and RULE sections is constructed using dot notation with a specific format. However, temporary variables, which will not affect the modeling, can be used. For more information, see Field structure.

### 3.11.2.5 | Field structure

#### Abstract

Learn more about the structure of the fields in the MODEL and RULE sections when creating Data Model Rules.

#### PREREQUISITE:

Data Model Rules requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Dataset Management, Parsing Rules, and Event Forwarding.

When creating Data Model Rules, each field used in the MODEL and RULE sections is constructed using dot notation using the following format:

```
xdm.<context>.[<compound>].<field>
```

Example 51.

- `xdm.<context>.[<compound>].<field>`

Example 52.

```
xdm.source.host.device_id
```

- `xdm.<context>.<field>`

Example 53.

```
xdm.source.ipv4
```



Part	Description
<context>	This is a composition of fields (<field>), either simple or <compound>, that are grouped together to form a logically coherent unit.
<compound>	This is a set of simple fields that are grouped together to form a meaningful group. For example, <code>subject</code> and <code>recipients</code> are part of the <compound> field called <code>email</code> .
<field>	This is a field that represents a primitive data type, such as a string or number or an array, or an IP address.

#### NOTE:

For more information on these data model fields, see Cortex XSIAM Data Model Schema.

#### Using ENUM fields

For fields of the `ENUM` type, you can map values from a predefined list of ENUMS. For example, the field `xdm.network.ip_protocol` is defined as `Enum.IP_PROTOCOL`, so you can assign it values such as `XDM_CONST.IP_PROTOCOL_TCP`. The full list can be found in the automatically suggested values for the relevant fields.

This syntax is not mandatory, and you can map any `STRING` value, but we recommend its use for consistency across all model mapping.

#### Example 54.

```
[RULE: common_ngfw_modeling]
alter xdm.source.ipv4 = json_extract_scalar(actor, ".$client_ip")
| alter xdm.network.ip_protocol = if(
    proto = 6, XDM_CONST.IP_PROTOCOL_TCP,
    proto = 11, XDM_CONST.IP_PROTOCOL_UDP,
    proto
);
```

### 3.11.2.6 | How to map authentication story events?

#### Abstract

Learn how to map authentication story events to the Cortex AgentIX Cortex Data Model.

#### PREREQUISITE:

Data Model Rules requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Dataset Management, Parsing Rules, and Event Forwarding.

Cortex AgentIX enables analytics to run on all mapped authentication data, which automatically creates authentication stories for Cortex Data Model (XDM) identity data. As a result, you need to map authentication events to the Cortex AgentIX XDM schema to build the authentication story. For a complete list of these fields, see XDM fields for mapping authentication events.

#### IMPORTANT:

This Feature focuses on authentication events related to SSO (Single Sign-On) and SaaS (Software-as-a-Service) application authentications. It does not cover internal authentication mechanisms such as Kerberos, NTLM, or traditional domain logon events generated by on-premise infrastructure.

#### PREREQUISITE:

Familiarize yourself with the Cortex Data model (XDM) schema for field definitions and naming conventions, see Cortex XSIAM Data Model Schema.

#### Mapping principles

When mapping your authentication data, follow these guidelines:

- **Prioritize conclusive events:** Focus on mapping events that clearly represent the final stage or result of an authentication process.
- **Exclude ambiguous or non-final steps from outcome decisions:** Intermediate or informational events should not be treated as indicators of success or failure.
- **Preserve context across the full authentication flow:** Include intermediate events to provide visibility into the process, while making it clear they are not final outcomes.
- **Normalize raw error and outcome data:** You must define explicit mapping logic between the raw event fields that contain outcome or error messages, such as `get_reason` and `debugdata_errorcode`, and the target XDM fields. This logic should normalize provider-specific strings or codes into a canonical format to support reliable detection and analytics across diverse sources.



Why follow the mapping principles?

- Prevents misclassification of failed sessions as successful.
- Avoids distorted behavioral baselines that can mask real attacks.
- Preserves full visibility of authentication flows without misleading analytics.

Third-party mapping examples

- DUO- SSO - Data Model Mapping
- Okta - SSO - Data Model Mapping

Mandatory XDM fields to map for authentication events

There are mandatory fields that you need map to the Cortex Data Model (XDM) schema to build authentication stories based on the mapped authentication data. For more detailed information on these fields, see XDM fields for mapping authentication events.

The following fields are mandatory to map:

- `xdm.source.port`
- `xdm.target.ipv4`
- `xdm.target.port`
- `xdm.network.ip_protocol`
- `xdm.source.ipv4`
- `xdm.event.type`
- `xdm.event.tags`
- `xdm.event.operation`
- `xdm.event.original_event_type`
- `xdm.auth.service`
- `xdm.event.outcome`
- `xdm.source.user.upn`

#### **IMPORTANT:**

To maximize the variety of issues that are retrieved based on the XDM authentication stories, we recommend that the following additional fields are populated: `xdm.target.resource_name`, `xdm.logon.type`, `xdm.source.user_agent`, and `xdm.source.host.device_category`. Should you decide to change the default XDM mappings, ensure that both the mandatory and recommended fields are populated and do not contain any empty values.

### 3.11.2.7 | Create Data Model Rules

Abstract

Cortex AgentIX includes an editor for creating Data Model Rules.

#### **PREREQUISITE:**

Data Model Rules requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Dataset Management, Parsing Rules, and Event Forwarding.

You can override rules or create your own rules using XQL and additional custom syntax that is specific to defining Data Model Rules. Once you edit a default data model mapping, you will no longer receive Marketplace updates.

Review the following:

- Data Model Rules editor views
- Data Model Rules file structure and syntax
- How to map authentication story events?

How to create Data Model Rules

1. In Cortex AgentIX, select Settings → Configurations → Data Management → Data Model Rules.
2. Select the Data Model editor view for writing your Data Model Rules.



You can select one of the following views:

- User Defined Rules: Leave the default view open and write your Data Model Rules directly in the editor.
- Both: Select this view to see the Data Model Rules editor as well as the default rules as you write your Data Model Rules.

3. Write your rules using XQL syntax and the syntax specific to Data Model Rules.

4. (Optional) Use XQL Search to test your Data Model Rules and review logs.

You can create queries on the data model. For more information, see [Create XQL query](#).

### 3.11.2.8 | Troubleshooting Data Model Rules

#### Abstract

Learn more about how to easily identify and resolve Data Model Rules errors in Cortex AgentiX.

#### PREREQUISITE:

Data Model Rules requires View/Edit RBAC permissions for Data Management (under Configurations à Data Management), which are the same permissions required for Dataset Management, Parsing Rules, and Event Forwarding.

To help you easily identify and resolve errors related to invalid Cortex Data Model (XDM) Rules, Cortex AgentiX provides the following:

- When an XDM query runs and one of the Data Model Rules is invalid, the invalid rule is automatically disabled and excluded from the query, and a warning is displayed.
- When a Data Model Rule is disabled, a message is added to your Cortex AgentiX console Notification Center. For more information about the Data Model Rules notifications, see [Data Model Rules notifications](#).
- The Data Model Rules editor displays an error icon and a message beside invalid Data Model Rules.
- An audit log is added to the Management Audit Log whenever a Data Model Rule becomes invalid, and when an invalid Data Model Rule becomes valid.

#### TIP:

To ensure you and your colleagues stay informed about Data Model Rules activity, you can also Configure notification forwarding to forward your Data Model Rules audit logs to an email distribution list or Syslog server. For more information about the Data Model Rules audit logs, see [Monitor Data Model Rules activity](#).

- When a rule is fixed, it is automatically enabled. User defined Data Model Rules are updated manually in the User Defined Rules editor. While default Data Model Rules are updated as part of a Marketplace package update, or a background change, such as an XQL content change.
- All Data Model Rules compilation errors are added to the `parsing_rules_errors` dataset.

#### Dataset for Data Model Rules Errors

All Data Model Rules compilation errors, such as syntax errors, missing arguments, and invalid regex, are saved to a dataset called `parsing_rules_errors`. This dataset also includes Parsing Rules errors. The following table describes the fields that are applicable to troubleshooting Data Model Rules errors when running a query in XQL Search for the `parsing_rules_errors` dataset in alphabetical order.

#### NOTE:

Since this dataset also contains Parsing Rules errors, some of the fields are irrelevant for Data Model Rules and aren't included in the table.

Read more...

Field	Description
CREATED_AT	Displays the timestamp when the error was generated.
ERROR_CATEGORY	Displays the category of the error, which for Data Model Rules errors is always Compile for compilation errors.
ERROR_MESSAGE	Displays the error message.
_ID	Displays the Rule ID that triggered this error.



Field	Description
RULE_TYPE	Displays the type of rule that triggered this error.
TARGET_DATASET	Displays the target dataset associated to the rule that triggered this error.
_TIME	Displays the timestamp when the error was generated.
XQL_TEXT	Displays the specific section of the Data Model Rule related to the error generated.

### 3.11.2.9 | Using data enrichment

#### Abstract

Learn about data-enriched fields and their limitations.

#### PREREQUISITE:

Data Model Rules requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Dataset Management, Parsing Rules, and Event Forwarding.

Cortex AgentiX automatically enriches your Cortex Data Model (XDM) data with additional information and context. Some examples of the types of data that are enriched include:

#### NOTE:

For a complete list of auto-enriched fields, see the Cortex Data Model Schema Guide.

- IP addresses are enriched with geolocation information.
- User data is normalized.
- If DSS exists, it is also enriched.

These enrichments are important for cyber analytics, rule detection, and investigations. Since these fields are enriched automatically by default, they do not have to be mapped manually in Data Model Rules. Note that enrichment is not performed when the input fields needed for enrichment are not available.

Enriched data is calculated by the system upon ingestion, and is saved for future queries. Keep in mind that some data may change over time, such as IP addresses that may change geolocation. Therefore, checking the same IP address in external systems at a later time might return a different geolocation result.

#### Overriding Data Enrichment

We do not recommend overriding enriched fields. However, if enriched fields are not desired, they can be overridden by mapping data to fields that are usually enriched.

Example 55.

```
[MODEL: dataset=okta_sso_raw]
| alter xdm.source.ip = actor->ip_address,
  xdm.source.location.country = actor->country,
  xdm.source.location.city = actor->geo.city;
```

When overriding enriched fields, ensure the following:

- The overridden data should be normalized.
- All relevant enriched fields should be overridden (for example, all location fields), and empty values should be filled with “unknown” (or with NULL, if calculated enrichments are desired). These actions will prevent data mismatch and conflicts.

#### IMPORTANT:

When manually mapping ASN fields that are enriched, such as `xdm.source.asn.as_number`, with other ISP and domain fields that are not enriched, such as `xdm.source.asn.isp` and `xdm.source.asn.domain`, it's possible to receive incorrect XDM query results due to the misalignment between the overridden enrichment and system enrichment fields.

#### Limitations



- Geolocation limitations
  - Some values will be NULL if the log country doesn't match the country detected by an external geolocation tool.
  - There might be discrepancies when some data come from the log and other data from the enrichment. For example, log country data versus enrichment longitude data.
- Data enrichment is not performed for EDR events.
- This feature is not supported in cold storage.

#### Backward compatibility

Data ingested by versions prior to Cortex AgentiX version 1.3 will not be enriched, because enrichment is calculated at the time of ingestion.

By default, enrichment is performed for NULL values only (non-NULL values are not overridden). Therefore, some existing mapping rules may need to be updated, in order to prevent mapping data to the enriched fields. Contact Customer Support for assistance with converting custom modeling rules and saved queries.

### 3.11.2.10 | Data Model Rules notifications

#### Abstract

Learn more about the notifications that are relevant for Cortex AgentiX Data Model Rules.

#### **PREREQUISITE:**

Data Model Rules requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Dataset Management, Parsing Rules, and Event Forwarding.

To help you monitor effectively your Data Model Rules, Cortex AgentiX sends notifications to your Cortex AgentiX console Notification Center.

Cortex AgentiX sends the following notification:

- Invalid Data Model Rules: Notifies when a Data Model Rule is invalid and will be excluded from `datamodel` queries.

To ensure you and your colleagues stay informed about Data Model Rules activity, you can also Configure notification forwarding to forward your Data Model Rules logs to an email distribution list or Syslog server. For more information about the Data Model Rules audit logs, see Monitor Data Model Rules activity.

### 3.11.2.11 | Monitor Data Model Rules activity

#### Abstract

Learn more about the monitored Cortex AgentiX Data Model Rules activities.

#### **PREREQUISITE:**

Data Model Rules requires View/Edit RBAC permissions for Data Management (under Configurations → Data Management), which are the same permissions required for Dataset Management, Parsing Rules, and Event Forwarding.

Cortex AgentiX logs entries for events related to the Data Model Rules monitored activities. Cortex AgentiX stores the logs for 365 days. To view the Data Model Rules audit logs, select Settings → Management Audit Logs.

To ensure you and your colleagues stay informed about Data Model Rules activity, you can Configure notification forwarding to forward your Data Model Rules audit logs to an email distribution list or Syslog server.

You can customize your view of the logs by adding or removing filters to the Management Audit Logs table. You can also filter the page result to narrow down your search. The following table describes the default and optional fields that you can view in the Cortex AgentiX Management Audit Logs table:

#### **NOTE:**

Certain fields are exposed and hidden by default. An asterisk (\*) is beside every field that is exposed by default.

Field	Description
Description*	Log message that describes the action.
Email	Email of the user who performed the action.



Field	Description
Host Name*	This field is not applicable for Data Model Rules logs.
ID	Unique ID of the action.
Reason	This field is not applicable for Data Model Rules logs.
Result*	The result of the action ( <code>Success</code> , <code>Fail</code> , or <code>Partial</code> )
Severity*	Severity associated with the log: <ul style="list-style-type: none"> <li>• <code>Critical</code></li> <li>• <code>High</code></li> <li>• <code>Medium</code></li> <li>• <code>Low</code></li> <li>• <code>Informational</code></li> </ul>
Timestamp*	Date and time when the action occurred.
Type* and Sub-Type*	Additional classifications of Data Model Rules logs (Type and Sub-Type): <ul style="list-style-type: none"> <li>• XDM Config:               <ul style="list-style-type: none"> <li>◦ Saving XDM mappings file: Indicates whenever a Data Model Rule is saved in the editor, the specific changes made to the Cortex Data Model (XDM) mappings. In addition, indicates whenever the changes weren't able to be saved.</li> <li>◦ Disabled: Indicates the Data Model Rule and associated dataset that are now disabled. This invalid rule is excluded from the query until the changes are made to fix the problem.</li> <li>◦ Enabled: Indicates the Data Model Rule and associated dataset that have been updated and are now enabled.</li> </ul> </li> </ul>
User Name*	Name of the user who performed the action.

### 3.11.3 | Manage Event Forwarding

#### Abstract

Save your ingested, parsed data in an external location by exporting your event logs to a temporary GCP storage bucket.

#### PREREQUISITE:

Event Forwarding requires View/Edit RBAC permissions for Data Management (under Configurations â Data Management), which are the same permissions required for Parsing Rules, Data Model Rules, and Dataset Management.

You can save your ingested, parsed data in an external location by exporting your event logs to a temporary storage bucket on Google Cloud Platform (GCP).

#### NOTE:

After exporting logs, you can download them from GCP for up to 14 days. The Pub/Sub subscription messages are available for 7 days.

Event forwarding has the following purposes:



- Compliance: You may have specific compliance requirements to retain logs in a separate, secure environment for long-term storage or auditing purposes.
- Long-term archive: The core function is to export the event logs that the tenant has ingested and parsed to a storage location outside of the XSIAM tenant. This provides you with a copy of the normalized and processed data.
- External analytics: Download the exported event logs for use in other security tools, data analysis platforms, or for offline forensic investigation.

You can forward the following events to GCP:

- Endpoints Event Forwarding

Forwards raw, high-fidelity security telemetry collected by EDR, including data from endpoints through the XDR Agent and cloud workloads (VMs, containers, or third-party EDRs). The exported logs are raw data, without any stories, and export a subset of the endpoint data without filtering or configuration options. For more information about the type of information forwarded, see [Endpoints Event Forwarding - included/excluded fields by event type](#).

**NOTE:**

Requires the Endpoints Event Forwarding add-on.

- GB Event Forwarding

Covers all other security data measured by daily ingestion volume (in Gigabytes). This includes non-endpoint logs such as firewall traffic, cloud audit logs, network flow logs, identity data, and general Syslogs from servers and devices. The exported logs are raw data, without any stories, and export all the data without filtering or configuration options.

**NOTE:**

Requires the GB Event Forwarding add-on

Use the Event Forwarding page to activate Event Forwarding, to retrieve the path and credentials of your external storage destination on GCP. Once this page is activated, Cortex AgentIX automatically creates the GCP bucket.

**IMPORTANT:**

Since data is aggregated and compressed, it may take up to two hours until the data is available in the forwarding bucket.

Upload to a temporary GCP storage bucket

Before you begin, ensure that you have the view/edit permission for Data Management. Instance Administrators have this permission by default.

1. Under Settings → Configurations → Data Management → Event Forwarding, activate one or more of the following:

- Enable GB Event Forwarding

2. Save your selection.

The Destination section displays the details of the GCP bucket created by Cortex AgentIX, where your data is stored for 14 days. The data is compressed and saved as a line-delimited JSON gzip file.

3. Access GCP Cloud Storage using a Service Account.

- a. Copy the storage path displayed.

- b. Generate and download the Service Account JSON WEB TOKEN, which contains the access key.

Save it in a secure location. If you need to regenerate the access token, Replace and download a new token. This action invalidates the previous token.

The token provides access to all your data stored in this bucket. It must be saved in a safe place.

Use the storage path and access key to manually retrieve your files or use an API for automated retrieval.

- c. Using the storage path and access key, retrieve your files manually or using an API.

- Authenticating as a service account
- Copying files and objects from GCP

4. (Optional) Use the Pub/Sub subscription to ensure reliable data retrieval without any loss.

- a. Copy the Pub/Sub subscription provided.

- b. Configure your application or system to receive messages from the Pub/Sub subscription.

Whenever a new file is added to the GCS bucket, a message is sent to the Pub/Sub subscription. The object path of the file in the bucket has the prefix `internal/`.

- c. Process the received message to initiate the download of the corresponding file.



## 3.12 | Dashboards and Reports

### Abstract

Learn about how to create and edit dashboards and reports.

View the Command Center and pre-defined dashboards, and create and edit dashboards and reports.

### 3.12.1 | About dashboards

#### Abstract

Learn more about dashboards, which help you to monitor system activity and security operations in your environment.

Dashboards help you monitor system activity and security operations in your environment. Each dashboard consists of widgets that summarize information about your tenant's activities in a graphical or tabular format, enabling you to effectively monitor your cases and overall activity in your environment.

When you sign in to Cortex AgentiX, your default dashboard is displayed. To change the displayed dashboard, you can select from the list of predefined and custom dashboards using the dashboard menu, or manage them centrally from the Dashboard Manager.

On each dashboard, you can see the selected Time Range on the right side of the header. To see the last updated status for each widget, hover over a widget and if required, Refresh the data. You can also select widget specific time frames from the menu on an XQL widget.

Predefined dashboard filters are displayed in the dashboard header. A filter icon on a widget indicates that the widget data is filtered. Hover over the icon to see details of the filters applied.

Click the dashboard menu to see additional actions, including the option to save the dashboard as a report template, set it as your default dashboard, and pause automatic dashboard refresh.

#### Types of dashboards

Cortex AgentiX provides the following types of dashboards:

- **Command Center dashboards:** These are system-provided dashboards that offer interactive, high-level overviews of system status, data ingestion, and security operations with built-in drilldowns. These dashboards are not editable. For more information, see [Command Center dashboards](#).
- **Predefined dashboards:** These are read-only dashboards tailored for common use cases and system setups to assist in both security operations and posture management. These dashboards provide immediate visibility into various aspects of your environment, including active threats, asset risk levels, and data ingestion health. You can create reports and custom dashboards that are based on predefined dashboards. For more information, see [Predefined dashboards](#).
- **Custom dashboards:** These are user-defined dashboards that provide the flexibility to design views according to your own specifications. You can base custom dashboards on the predefined dashboards or create a new dashboard from scratch, and save your custom dashboards as reports. Ownership and access for these dashboards are managed through the Share or Manage Access options in the Dashboard Manager. For more information, see [Build custom dashboards and reports](#).

#### Manage your dashboards

You can view and manage all predefined and custom dashboards from the Dashboard Manager. The actions you can perform on a dashboard depend on its visibility status and the permissions granted to your role.

#### **IMPORTANT:**

The ability to create, edit, or share custom dashboards is governed by access management. If certain options are unavailable, contact your administrator. For more information, see [Manage access to custom dashboards](#).

From the Dashboard Manager, you can take the following actions:



- **Manage visibility:** Custom dashboards can be set to either Public or Restricted.
  - Public: Visible to all users with appropriate role permissions to view dashboards.
  - Restricted: Visible only to the dashboard Owner and the specific users or user groups who have been granted explicit access through sharing.
- **Create, edit, and delete custom dashboards:** You can build new dashboards to suit your specific needs. While you cannot edit predefined dashboards, you can save them as a new custom dashboard to use as a starting template. Restricted dashboards can be viewed and edited by the Owner and any users or user groups granted explicit access. Yet, only the Owner (the creator) or an administrator can delete a custom dashboard.
- **Manage access:** If permitted by your administrator, you can share custom dashboards you own with other users or user groups, granting them either Viewer or Editor permissions. Dashboards can only be shared with users and user groups and not API keys.
- **Select your default dashboard:** Set any available dashboard as the primary view when you log in.
- **Create report templates:** Use a dashboard layout as the basis for a scheduled report.
- **Import and export dashboards:** Administrators can import and export dashboards in a JSON format, which enables transferring configurations between environments for onboarding, migration, backup, and sharing. It's also possible to bulk export and import multiple dashboards at a time.

**NOTE:**

- Dashboards that are based on custom infrastructure cannot be exported.
- When importing a dashboard that already exists in the system, the imported dashboard overwrites the existing dashboard. To avoid overwriting the existing dashboard, duplicate and rename the existing dashboard before importing the new dashboard.

#### Dashboard sharing icons

The following icons in the Dashboard Manager (under the Source column) help you identify the security access and status of your dashboards:

- : A Restricted custom dashboard you created (Owner) that is not currently shared with anyone else.
- : A custom dashboard you created that is currently shared with other users or user groups.
- : A custom dashboard created by another user that has been shared with you (either individually or through a user group).
- : A standard system dashboard provided by Palo Alto Networks. These are always Public and can't be deleted, or have their ownership transferred.

#### 3.12.1.1 | Command Center dashboards

##### Abstract

Learn more about Command Center dashboards, which are system-provided dashboards.

Command Center dashboards are system-provided dashboards that offer interactive, high-level overviews of your security operations, data ingestion, and system status. From the Command Center dashboards, click on elements of interest to drill down to additional dashboards and associated pages.

These dashboards are read-only and are provided by Palo Alto Networks to ensure instant visibility into your environment. Because these are system-provided dashboards, they are always Public and visible to all authorized users. Palo Alto Networks also provides regular updates to these dashboards to ensure they reflect the latest system capabilities.

**NOTE:**

- Access to these dashboards requires RBAC permissions under Dashboards & Reports. The Dashboards component must be Enabled in your role and requires View permissions for Command Center Dashboards. If certain options are unavailable, contact your administrator. For more information, Manage access to custom dashboards.
- The dashboards are available in dark mode only. They are not editable, and you can't create dashboard templates or reports from them.
- Some of the dashboard's animations are not fully supported by the Safari web browser. We recommend that you view the dashboard with an alternative web browser.

#### 3.12.1.1.1 | AgentiX Command Center

##### Abstract

See a dynamic overview of how AgentiX capabilities are utilized across your organization.

The AgentiX Command Center dashboard enables you to view how Cortex AgentiX automation powers your SOC. Use the AgentiX Command Center to identify key areas of utilization and find opportunities to drive greater adoption and optimize your automation strategies.

The AgentiX Command Center includes the following:



- Pre-configured Triggers: Playbooks that were triggered by automation rules. Clicking on Pre-configured Triggers shows how many rules were triggered and the top automation rules. For more information about rule-based automation, see Create an automation rule.
- User Prompts: Clicking on User Prompts shows the number of users who created the user prompts and usage over the past seven days.
- Agent Grid: The grid shows the five AI agents that users engaged with most frequently in the past seven days. Clicking on an agent opens the agent card, which provides more details about the roles that can access the agent and the actions available to the agent. Other Agents represent all other enabled agents, such as system and custom agents. Clicking on Other Agents brings you to the Agents Hub, where you can view all agents.
- Key performance indicators: Display automation-related metrics. Click on a metric to open a related page or pop-up with more information.

You can also open the Cortex Agentic Assistant chat to start an investigation, view top external services, and see how many plans require intervention due to pending user action, permissions, integration gaps, and other errors.

Unless you have specified an alternative dashboard, this is the default dashboard in Cortex AgentiX.

#### **NOTE:**

- Access to the dashboard requires RBAC View permissions for Dashboards & Reports and Command Center Dashboards. Non-admin users who have these permissions can view shared system and custom agents as well as their own agents, listed under Other Agents. They cannot see other users' agents.
- The dashboards are available in dark mode only. They are not editable, and you can't create dashboard templates or reports from them.
- Some of the dashboard's animations are not fully supported by the Safari web browser. We recommend that you view the dashboard with an alternative web browser.

#### 3.12.1.2 | Predefined dashboards

##### Abstract

Learn more about predefined dashboards, which are out-of-the-box dashboards provided by Palo Alto Networks.

Cortex AgentiX provides predefined dashboards that display widgets tailored to the dashboard type. The dashboards can help you monitor different aspects of your environment. To access your default dashboard, select Dashboards & Reports → Dashboard. From the dashboard header, a drop-down menu lists all available predefined and custom dashboards. The available dashboards depend on your license type.

Since predefined dashboards are system-managed and cannot be edited or deleted, you can create your own copy of a dashboard by selecting Save as new. This allows you to edit the widgets and configuration from your own custom version while preserving the original one.

##### Access and visibility

Since these dashboards are provided directly by Palo Alto Networks, they follow a standard access model:

- **Always Public:** These dashboards are always Public and visible to all authorized users. Unlike custom dashboards, you do not need to manage a list of who can see them.
- **Role permissions:** Your visibility and access to these dashboards are set by your administrator through your user role. If you are unable to view specific dashboards or perform certain actions, contact your administrator to ensure your role permissions are configured correctly.
- **Data scoping:** While the dashboard structure is public, the data you see within the widgets is automatically filtered based on your authorized data scope. For example, you will only see information for the asset groups you are permitted to view.

##### Available predefined dashboards

Dashboard Name	Description
Automation Insights	Provides a high-level overview of automation, focusing on issues automatically closed and execution trends.
Cortex Cloud Command Center	A central hub that provides a prioritized high-level summary of cloud accounts, provider scanning health, and asset distribution across all cloud providers, tracking progress over time with 90-day trends for Threat and Posture cases. For more information, see Cortex Cloud Command Center.
Cloud Security Operations	Provides an overview of your cloud security operations dashboard helps you rapidly assess your security posture and resolve issues with the largest impact. For more information, see Cloud Security Operations.
My Dashboard	Provides an overview of the cases and MTTR for the logged-in user.



Dashboard Name	Description
Security Admin	Provides an overview of the cases in your organization and the status of resolved and overdue cases.
Threat Intel Management	<p>Provides information about malicious or suspicious indicators in cases.</p> <p><b>LICENSE TYPE:</b></p> <p>Requires the Cortex XSIAM Premium license or any other XSIAM license with the Threat Intel Management (TIM) add-on to use this feature.</p>
Troubleshooting Instances	Provides a detailed view of command and execution errors at the instance level, helping diagnose and resolve issues with specific integrations.
Troubleshooting Playbooks	Provides the ability to identify and resolve issues with playbooks and tasks through focused error analysis and runtime metrics.

### 3.12.2 | Reports

#### Abstract

Create, edit, and customize reports in Cortex AgentIX. Schedule reports with Cron expressions.

Reports contain statistical data in the form of widgets, which enable you to analyze data from inside or outside Cortex AgentIX, in different formats such as graphs, pie charts, or text from information. After generating a report, it also appears in the Reports tab, so you can use the report again.

#### 3.12.2.1 | Report templates

##### Abstract

View, import, export, create, and modify report templates

On the Report Templates page, you can view, delete, import, export, create, and modify report templates. You can also select and generate multiple reports.

#### 3.12.2.2 | Run or schedule reports

##### Abstract

You can run reports that are based on dashboard templates, or you can create reports from scratch.

You can generate reports using pre-designed dashboard templates, or create custom reports from scratch with widgets from the Widget Library. You can also schedule your reports to run regularly or just once. All generated reports are saved under Dashboards & Reports → Reports.

To take actions on existing report templates, go to Dashboards & Reports → Report Templates. On this page you can also import and export report templates in a JSON format, which enables you to transfer your configurations between environments for onboarding, migration, backup, and sharing. You can bulk export and import multiple report templates at a time.

#### NOTE:

- Report templates that are based on custom infrastructure cannot be exported.
- If you import a report template that already exists in the system, the imported template will overwrite the existing template. If you do not want to overwrite the existing template, duplicate and rename the existing template before importing the new template.
- You can also quickly create a report template from an existing dashboard by right-clicking the dashboard in the Dashboard Manager and selecting Save as report template.

##### Access and visibility

- **Role permissions:** Your ability to run and schedule reports is set by your administrator through your user role.
- **Widget visibility in reports:** Reports can include both Public and Restricted widgets. When you add a widget to a report template, the report will display the data from that widget to all report recipients, regardless of whether the recipients have access to that specific widget in their own Widget Library.
- **Data scoping:** Reports pre-fetch data based on the scope of the user who last saved the template. Ensure the template creator has the appropriate data permissions to provide the intended results for the report recipients.



## Dashboard sharing icons

The following icons in the Dashboard Manager (under the Source column) and in the Dashboards page header menu help you identify the security access of your dashboards:

- A Restricted custom dashboard you created (Owner) that is not currently shared with anyone else.
- A custom dashboard you created that is currently shared with other users or user groups.
- A custom dashboard created by another user that has been shared with you (either individually or through a user group).
- A standard system dashboard provided by Palo Alto Networks. These are always Public and can't be deleted, or have their ownership transferred.

## Run a report based on a dashboard

You can generate a report based on an existing dashboard.

1. Select Dashboards & Reports → Dashboard Manager.
2. Right-click the dashboard you want to use, and select Save as report template.
3. Enter a unique name for the report and an optional description, and click Save.
4. Select Dashboards & Reports → Report Templates.
5. Locate your new report template, right-click it, and select:
  - Generate Report: To run the report immediately.
  - Edit: To modify parameters or configure a schedule.
6. After your report completes, you can download it from the Dashboards & Reports → Reports page.

## Create a new report template

You can base your report on an existing template, or you can start with a blank template.

1. Select Dashboards & Reports → Report Templates, and click New Template.
2. Enter a unique name for the report and an optional description.

### NOTE:

The report name and description will be displayed in the report header and are not editable during customization.

3. Under Data Timeframe, select the duration for the report. Custom time frames are limited to one month.
4. Choose a Report Type (a built-in template or blank) and click Next.
5. Customize your report.

Cortex AgentiX offers mock data to help you visualize the data's appearance (default view). To see how the report would look with real data in your environment, switch to Real Data. Select the Preview in A4 icon to see how the report is displayed in an A4 format.

6. Add or remove widgets to the report. From the widget library, drag widgets on to the report layout. You can add both standard system widgets and your custom (Public or Restricted) widgets.

### NOTE:

- For agent-related widgets, you can apply an endpoint scope to refine the displayed data to only show results from specific endpoint groups.  
Select the menu on the top right corner of the widget, select Groups, and select one or more endpoint groups.
- For case-related widgets, you can refine the displayed data to only show results from cases that match a case starring configuration. A purple star indicates that the widget is displaying only starred cases. For more information, see Case starring.

7. (Optional) Add filters to the report. Adding filters and inputs to the report gives you the flexibility to filter report data based on default values that you define.

If you selected a report template with default filters, the filters are displayed at the top of the dashboard. To edit the filters, click Add Filters & Inputs.

You can configure basic filters that provide predefined static values, as explained in the following steps. Alternatively you can define dynamic filters that are based on predefined parameters in custom XQL widgets, as explained in Configure filters and inputs for custom XQL widgets.

- a. Click Add Filters & Inputs to define parameters for the report data.
- b. On the FILTERS & INPUTS panel, select a parameter for which to configure a filter.
- c. Under Value, select one or more filter values.



If no values are selected, the filter name shows an error symbol and you cannot save the filter.

d. Add more filters as required. You can drag the filters to change the priority.

e. Click Save Filters & Inputs.

8. Click Next.

9. Configure the report execution:

- Generate now: To run a single instance immediately.
- Schedule: Define a recurring timeframe for automatic generation.

10. (Optional) Configure Email Distribution or Slack recipients for the PDF.

**NOTE:**

To send reports to Slack, Slack must be configured as an external application in Cortex AgentiX. For more information, see [Integrate Slack for outbound notifications](#)

11. (Optional) Select Attach CSV to include raw data from XQL widgets.

From the menu, select one or more of your custom widgets to attach to the report. The CSV files of the widgets are attached to the report along with the report PDF. Depending on how you selected to send the report, the CSV file is attached as follows:

- Email: Sent as separate attachments for each widget. The total size of the attachment in the email cannot exceed 20 MB.
- Slack: Sent within a ZIP file that includes the PDF file.

12. Click Save Template.

13. After your report completes, you can download it from the Dashboards & Reports â Reports page.

In the Name field, icons indicate the number of attached files for each report. Reports with multiple PDF and CSV files are marked with a zip icon. Reports with a single PDF are marked with a PDF icon.

Configure the notification rule for a failed report

You can receive an email or send a notification to a syslog server if a report fails to run due to a timeout or fails to upload to the GCP bucket.

1. Under Settings â Configurations â General â Notifications, click + Add Forwarding Configuration.
2. Enter a name and a description for your rule, and under Log Type, select Management Audit Logs.
3. Use a filter to select the Type as **Reporting**, Subtype as **Run Report**, and Result as Fail.
4. Enter a distribution list to receive notifications by email or select a syslog server.
5. Click Next.
6. Review settings and click Create.

### 3.12.3 | Build custom dashboards and reports

#### Abstract

Custom dashboards and reports can support your day-to-day operations by providing options that are tailored to your unique workflow.

You can create custom dashboards and reports that are tailored to your unique workflow and support your day-to-day operations. With custom dashboards and reports, you have the flexibility to base your dashboard or report on an existing template, or build a new template from scratch.

#### Access and visibility

Your ability to view and manage dashboards and reports is based on your user permissions and the visibility status of the specific dashboard:

- **Dashboard Manager:** This is the central repository for your visualizations. You can only access dashboards where you are the Owner, dashboards that have been explicitly shared with you (or your user group), or dashboards marked as Public.
- **Role requirements:** To create or manage dashboards, your administrator must Enable the Dashboards component in your user role. The specific actions you can take, such as creating new dashboards or editing public ones, depend on the sub-permissions assigned to your role. For more information, see [Manage access to custom dashboards](#),

#### Dashboard components

Dashboards and reports are built from widgets. You can drag any widgets from the Widget Library on to your dashboard or report and arrange them. Cortex AgentiX provides predefined widgets for you to use. In addition, you can create custom widgets that are built on Cortex Query Language (XQL) queries or custom scripts and provide the flexibility to query specific data, and select the graphical format you require (such as table, line graph, or pie chart).



- **Widget Library:** The Widget Library is the central repository for predefined and custom widgets and is intended solely for browsing and selecting widgets to add to a dashboard.
- **Widget visibility:** Those with access to the Widget Library will only see widgets in the Widget Library that are Public or widgets that were created by you (Restricted). You cannot see Restricted widgets created by other users in the Widget Library, even if you have access to a dashboard containing those widgets.
- **Inherited access:**
  - If a custom dashboard is shared with you, you can see all widgets contained within that dashboard, but you will never have permission to view or edit those specific widgets in the Widget Library (unless you are the owner, administrator, or it's a public widget).
  - If you don't have permission to edit a widget, create a copy of the widget.

### 3.12.3.1 | Build a custom dashboard

#### Abstract

Build customized dashboards to display and filter the information that is most relevant to you.

Build customized dashboards to display and filter the information that is most relevant to you. You can build custom dashboards based on predefined dashboard templates, or you can build a new dashboard from scratch.

By default, all new custom dashboards are Restricted and visible only to you (the Owner). Once created, you can use the Dashboard Manager to share the dashboard with specific users or groups, or make it Public to all authorized users.

1. Select Dashboards & Reports → Dashboard Manager → New Dashboard, or in the Dashboard Manager right-click an existing dashboard and select Save as new.
2. In the Dashboard Builder, under Dashboard Name, enter a unique name for the dashboard.
3. Under Dashboard Type, choose a built-in dashboard template or a blank template, and click Next.
4. (Optional) Change the toggle to Real Data to see how the dashboard looks with actual data from your environment.
5. Add widgets to the dashboard. From the Widget Library, drag widgets on to the dashboard.

#### **NOTE:**

- For case-related widgets, you can refine the displayed data to only show results from cases that match a case starring configuration. A purple star indicates that the widget is displaying only starred cases. For more information, see Case starring.

6. (Optional) Configure fixed filters and inputs.

#### What are fixed filters and inputs?

Add fixed filters to your dashboard to provide dashboard users with useful dashboard filters that are based on predefined or dynamic input values. Any defined filters are displayed in the dashboard header.

Fixed filters are based on XQL widgets with dynamic parameters. If a dashboard contains these widgets, the Add Filters & Inputs option is displayed. For more information, see Configure filters and inputs for custom XQL widgets.

7. (Optional for dashboards with custom XQL widgets) Configure dashboard drilldowns.

#### What are dashboard drilldowns?

Add drilldowns to your dashboard to provide interactive data insights when clicking on data points, table rows, or other visualization elements.

Dashboard drilldowns are based on XQL widgets. To add a drilldown to an XQL widget, click on the widget menu, and select Add drilldown. For more information, see Configure dashboard drilldowns.

8. Under Time Range, set a time range for your dashboard.

By default, the widgets use the dashboard timeframe. You can change the widget timeframe from the widget menu.

9. When you have finished customizing your dashboard, click Next.

10. (Optional) To set the custom dashboard as your default view when logging in, select Define as default dashboard.

11. Click Generate to complete your dashboard.

### 3.12.3.2 | Manage your Widget Library

#### Abstract

Create, search, and view custom widgets in Cortex AgentiX, or use predefined widgets.



The Widget Library is the central repository where you can view, create, and manage all predefined and custom widgets. All of your predefined and custom widgets are available in the Widget Library under Dashboards & Reports â†’ Widget Library.

From the Widget Library, you can select widgets to add to your dashboards or reports, or use existing widgets as templates for new ones. You can search for a widget by name, and use the filter to view only specific widget types. For example, you can filter for all widgets that use a certain chart type, or only view custom widgets you have created.

#### Access and visibility

Your ability to see and use widgets in the Widget Library is based on your user permissions and the visibility status of the widget:

- **Public widgets:** These include all predefined system widgets and any custom widgets that have been marked as Public. These widgets are visible to all users in the Widget Library who have the appropriate permissions to view dashboards and widgets.
- **Restricted widgets:** These are custom widgets that are visible only to the Owner (the creator) within the Widget Library.

#### NOTE:

Unlike dashboards, individual widgets cannot be shared with other users directly from the Widget Library. Yet, if you add a Restricted widget to a dashboard and then share that dashboard, other users who have access to the dashboard will also be able to see that specific widget.

#### Widget management actions

From the Widget Library you can take the following actions:

- **Create custom widgets based on XQL search queries:** Build new widgets from scratch using Cortex Query Language (XQL). By default, all new custom widgets you create are Restricted and visible only to you. For more information see [Create custom XQL widgets](#).
- **Search for custom and predefined widgets:** Widgets are grouped by category for easier discovery.
- **Edit or delete custom widgets:** You can modify or remove custom widgets that you own.

#### NOTE:

Any dashboards or reports that include the widget are affected by the changes.

- **Manage visibility:** If you are the Owner of a custom widget, you can change its status from Restricted to Public. This makes the widget available in the Widget Library for all other users who have the required permissions. For more information, see [Manage access to custom dashboards](#).

### 3.12.4 | Fine-tune dashboards and reports

#### Abstract

Fine tune your custom dashboards and report by adding custom XQL widgets, fixed filters and inputs, and dashboard drilldowns.

You can fine-tune your custom dashboards and reports to tailor them to suit your specific needs.

The following dashboard features enhance the functionality of your dashboards, by refining the data displayed and enabling dashboard users to filter and manipulate the displayed data. Click on the tabs to see each feature.

#### Custom script widgets

Build custom widgets based on scripts, to display data from third-party systems. These widgets can display pie, column, and line charts, as well as single value results. A single value result can, optionally, be presented as a time duration. By default, new custom script widgets are Restricted and visible only to the creator in the Widget Library.

#### Custom XQL widgets

Personalize the information that you display on your custom dashboards and reports by creating custom XQL widgets and adding them to your dashboard. By default, new custom XQL widgets are Restricted and visible only to the creator in the Widget Library. These widgets can query specific information that is unique to your workflow, and define the graphical format you require (such as table, line graph, or pie chart). In addition, you can add variables to your custom XQL widget that provide dynamic input values for dashboard filters.

#### Fixed filters and inputs

Enable dashboard users to alter the scope of a dashboard by selecting from predefined or dynamic input values.

#### Dashboard drilldowns

Enable dashboard users to access interactive data insights when clicking on data points in widgets. Drilldowns can trigger contextual changes on the dashboard, or they can link to an XQL search, a custom URL, another dashboard, or a report. Users can hover over a widget to see details about the drilldown, and click a value to trigger the drilldown.

### 3.12.4.1 | Create a custom widget using a script

#### Abstract



Create a custom script based widget. Use custom widgets in dashboards and reports.

You can use scripts in custom widgets to create dynamic widgets for more complex calculations and to present data from third-party systems. For examples of creating widgets using scripts, see [Script-based widget examples](#).

Before creating a script-based widget in the Widgets Library, you need to create or upload the script to the Scripts page. In the Widgets Library, you can change elements of the visual presentation. Because these widgets can contain unique logic or sensitive data queries, they are now managed as individual items with specific access rules.

#### Access and permissions

The ability to create and manage script-based widgets is determined by your role permissions and the widget's visibility:

- **Creation rights:** To create a script-based widget, your user role must allow you to create scripts and build dashboards. These permissions are set by your administrator. For more information, see [Manage access to custom dashboards](#).
- **Widget Library visibility:**
  - Restricted: New script widgets are Restricted by default. They are only visible to you within the Widget Library.
  - Public: If you want other users to be able to find and reuse your script in their own dashboards, the widget must be set to Public.

#### How to create a script-based widget

1. **Create the script:** Select Investigation & Response → Automation → Scripts. You can upload an existing script or create a new one. Cortex AgentX supports JavaScript, Python and PowerShell. You can create a script for one of the following chart types:

- Pie
- Column
- Line
- Single Value

2. **Configure for the Widget Library:** In the Script Settings, add the widget tag to the script. This tag ensures the script is recognized as a visualization tool and becomes available in the Widget Library.

3. **Create the Custom Script Widget:** Select Dashboards & Reports → Widget Library, click Create custom widget, and select Script.

4. **Define the widget properties:**

- Name and Description: Give your widget a clear name so you can identify it later in the Widget Library.
- Script: Select the script you created in step 1 from the list.

**NOTE:**

If you have added arguments to the script, these appear when creating a widget.

5. **Set visibility:** Use the Public widget toggle to determine how the widget appears in the Widget Library. Leave it unselected (default) to keep the widget Restricted (visible only to you) or select it to make the widget Public (visible to all users with Widget Library access).

6. **Preview and save:** Run a preview to ensure the script executes correctly and displays the data as intended, then click Save.

7. **Configure the display (Chart Editor):** Use the Chart Editor to choose the graph type and the subtype, and to enable or disable the graph legend.

**NOTE:**

Available options are Pie, Column, Line, and Single Value.

To display the result of the script as a time duration, choose the graph type Single Value and enable Show as Time. You can then select the Time Unit (millisecond, second, minute, or hour) and the Display format.

8. **Add to reports or dashboards:** Once saved to the Widget Library, you can add this script-based widget to any custom dashboard or include it when building a report template.

#### Managing existing script widgets

You can manage your script widgets directly from the Widget Library:

- **Reusability:** Any widget marked as Public can be searched for and added to any dashboard by other users.
- **Editing:** Only the person who created the script widget (the owner) or an administrator can modify the underlying script code in the Automation section.

#### 3.12.4.1.1 | [Script-based widget examples](#)

#### Abstract

Create script based widgets based on scripts for reports and dashboards in Cortex AgentX.



You can use script-based widgets to perform calculations on and visualize third-party data.

#### NOTE:

Add the widget tag in the script settings to make the script available for use in script-based widgets. For more information, see [Create a script](#).

The following are sample Python scripts for the graph types Single Value, Pie, Line, and Column.

##### Single value

This example shows how to use a script with an API call to return a single value in a widget. Use this example to build your own script that pulls in third-party data to display a single value.

#### NOTE:

If your script returns a time duration, configure the widget with the graph type Single Value and enable Show as Time..

Example:

```
import requests

def main():
    api_key = 'PUTYOURKEYHERE'
    symbol = 'PANW'
    api_url = f'https://www.alphavantage.co/query?function=GLOBAL_QUOTE&symbol={symbol}&apikey={api_key}'

    response = requests.get(api_url)
    data = response.json()

    price_str = data['Global Quote']['05. price']
    price_int = int(float(price_str))

    return_results(price_int)

if __name__ in ('__main__', '__builtin__', 'builtins'):
    main()
```

##### Pie, Line, or Column Chart

Example 1

The following example script creates random, mock data to simulate a stock price fluctuating over a short period of time. Use this example to build your own script that brings in third-party data and display trends using a pie, line, or column chart.

```
import random
import json
from datetime import datetime, timedelta

def main():
    chart_data = []
    start_time = datetime.strptime("13:00", "%H:%M")

    # Start the price at a realistic value
    current_price = 202.0

    # Simulate 50 data points
    for i in range(50):
        # Generate a time label in 1-minute jumps
        time_label = (start_time + timedelta(minutes=i)).strftime("%H:%M")

        # Create the data point for the chart
        data_point = {
            "name": time_label,
            "data": [int(current_price)],
            "groups": []
        }
        chart_data.append(data_point)

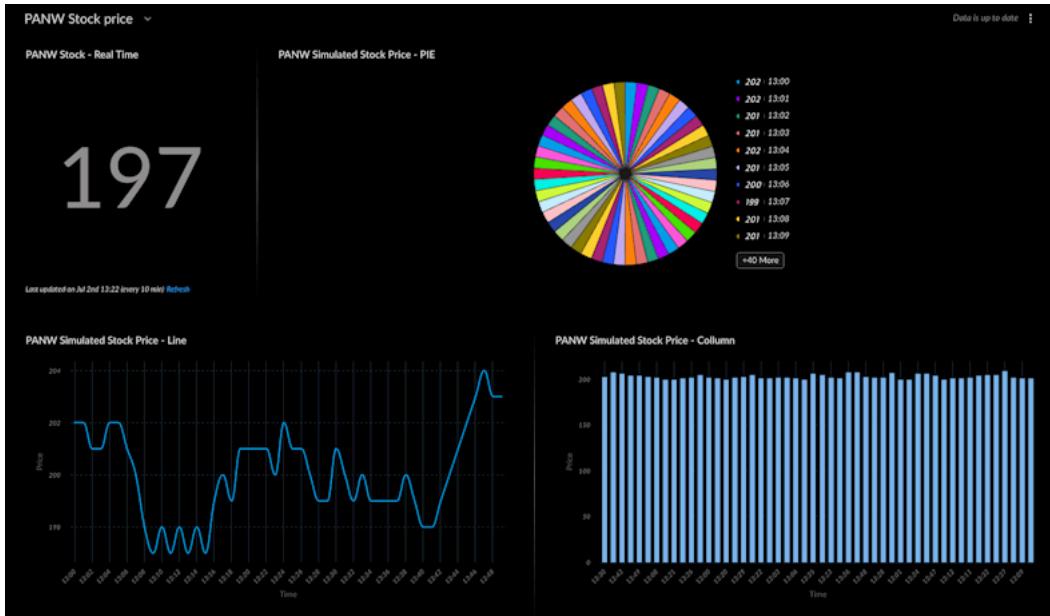
        # Simulate the next price by adding a small change to the current price
        price_change = random.uniform(-1.5, 1.5) # A small drift up or down
        current_price += price_change

    # Return the data formatted exactly as in your working script
    return_results({
        "Type": 1,
        "ContentsFormat": "json",
        "Contents": json.dumps(chart_data)
    })

if __name__ in ('__main__', '__builtin__', 'builtins'):
    main()
```

When used in a widget:





Example 2

The following example script generates simulated data representing the count of security incidents (or other events) broken down by severity level for each day of the week (Monday to Friday). Use this example to build your own script to create a stacked column chart. Configure the widget with graph type Column subtype Stacked.

```
import json
import random

def main():
    chart_data = []
    days = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday"]
    severities = ["Critical", "High", "Medium", "Low", "Info"]

    for day in days:
        groups_list = []
        daily_total = 0

        for severity in severities:
            count = 0
            if severity == "Critical":
                count = random.randint(0, 5)
            elif severity == "High":
                count = random.randint(5, 15)
            elif severity == "Medium":
                count = random.randint(10, 25)
            elif severity == "Low":
                count = random.randint(20, 50)
            else:
                count = random.randint(5, 30)

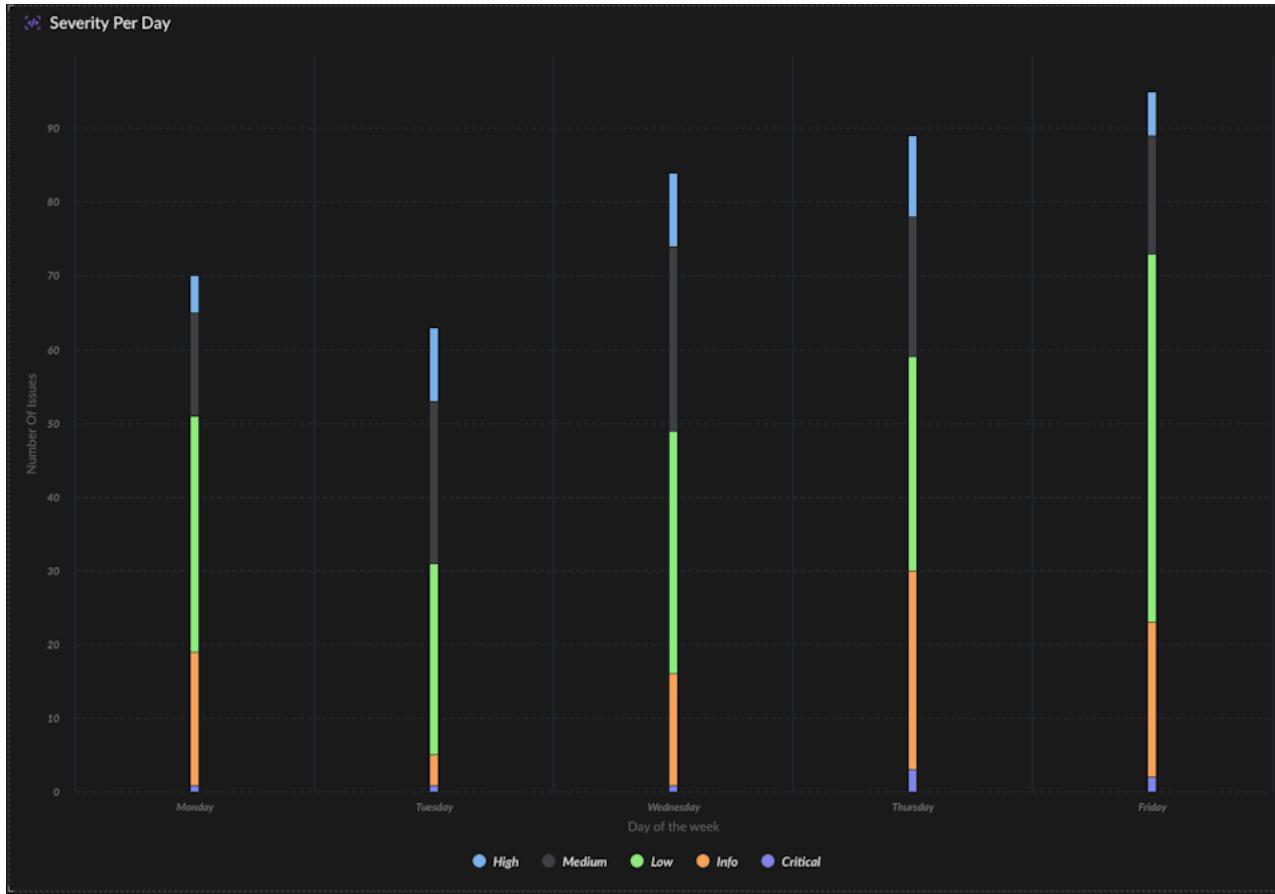
            daily_total += count
            groups_list.append({"name": severity, "data": [count]})

        chart_data.append({
            "name": day,
            "data": [daily_total],
            "groups": groups_list
        })

    return_results({
        "Type": 1,
        "ContentsFormat": "json",
        "Contents": json.dumps(chart_data)
    })
}
```

When used in a widget:





### 3.12.4.2 | Create a text widget

#### Abstract

Learn more about adding text widgets to dashboards for context, instructions, or additional information alongside data visualizations.

You can use a widget to display formatted text in dashboards and reports. You can also use scripts in custom widgets to create dynamic widgets for more complex calculations and to present data from third-party systems. Text widgets specifically allow you to use Markdown or HTML to include descriptive text, links, or images. You can create a text widget directly within a specific dashboard or build one in the Widget Library for reuse across your environment.

#### Access and visibility

- Role permissions:** Your visibility and access to these widgets are set by your administrator through your user role. For more information, see Manage access to custom dashboards.
- Visibility settings:** By default, text widgets are Restricted. They are visible to you in the Widget Library and on any dashboards you have shared. If you want other users to be able to find and reuse your text widget from the Widget Library, you must set it to Public.

#### Method 1: Add a text-based widget to a dashboard

1. Select Dashboards & Reports â Dashboard Manager.
2. Initiate the Dashboard Manager editor by taking one of the following actions:
  - To create a new dashboard: Click New Dashboard, enter a name, description, and choose a layout.
  - To edit an existing dashboard: Locate your custom dashboard, right-click it, and select Edit.
3. Click Next.
4. From the Widgets Library, drag the Free Text widget to your dashboard.
5. Add your custom text to the Free Text widget.
6. (Optional) To enable Markdown, click the vertical menu for the widget and select Use Markdown.
7. Click Next .
8. Choose your dashboard options and Generate dashboard.

#### Method 2: Add a text-based widget to a report template



**NOTE:**

From the Dashboard Manager, you can right-click an existing dashboard and choose Save as report template. You can also add text-based widgets directly to new or existing report templates, as described below.

1. Select Dashboards & Reports → Report Templates.
2. Initiate the Report Builder editor by taking one of the following actions:
  - **To create a new report template:** Click New Template, enter a name, description, timeframe, and either choose a report template type or build a new one from scratch.
  - **To edit an existing report template:** Locate your custom report template, right-click it, and select Edit.
3. Click Next.
4. From the Widgets Library, drag the Free Text widget to your dashboard.
5. Add your custom text to the Free Text widget.
6. (Optional) To enable Markdown, click the vertical menu for the widget and select Use Markdown.
7. Click Next .
8. Choose your report builder options and Save Template.

#### 3.12.4.3 | Create custom XQL widgets

##### Abstract

You can create custom XQL widgets based on a Cortex Query Language (XQL) query, and add parameters that you can configure as fixed filters or dashboard drilldowns.

With custom Cortex Query Language (XQL) widgets you can personalize the information that you display on your custom dashboards and reports. You can build widgets that query specific information that is unique to your workflow, and define the graphical format you require, such as table, line graph, or pie chart.

All of your predefined and custom XQL widgets are available in the Widget Library. From the Widget Library, you can browse all widgets by category, create new XQL widgets, and edit and delete existing XQL widgets.

##### Access and visibility

- **Role permissions:** Your visibility and access to these widgets are set by your administrator through your user role. For more information, see Manage access to custom dashboards.
- **Visibility settings:** By default, custom XQL widgets are Restricted. They are visible to you in the Widget Library and on any dashboards you have shared. If you want other users to be able to find and reuse your widget from the Widget Library, you must set it to Public.
- **Inherited access and reuse:**
  - **Dashboards:** If you add a Restricted XQL widget to a dashboard and share that dashboard, authorized viewers and editors of that dashboard can see the widget data even if they cannot see the widget in their own Widget Library.

##### How to create a custom XQL widget

1. Select Dashboards & Reports → Widget Library, click Create custom XQL widget, and select XQL.
2. Enter a widget name and an optional description.
3. Set the visibility for the widget.

Use the Public widget toggle to determine how the widget appears in the Widget Library. Leave it unselected (default) to keep the widget Restricted (visible only to you) or select it to make the widget Public (visible to all users with Widget Library access).

4. Define an XQL query that searches for the data you require.

Select XQL Helper to view commonly used commands with example syntax. For more information, see How to build XQL queries.

**TIP:**

You can create a generic dashboard for multiple views of the same dataset by using an asterisk (\*) when defining the dataset in the XQL widget as `dataset = <dataset_name>*`. The placement of the asterisk in the dataset name ensures that any view containing this prefix text is displayed in the results.

**Example 56.**

The dataset in a query is defined as:

```
dataset = amazon_aws_raw*
```



If there are multiple datasets that begin with `amazon_aws_raw` in their name, such as `amazon_aws_raw_eu_view` and `amazon_aws_raw_us1_view`, these views will be included.

5. Select Preview to review the search results.

**NOTE:**

Cortex Query Language (XQL) queries generated from the Widget Library do not appear in the Query Center. The results are used only for creating the custom widget.

6. (Optional) Add parameters to the query to enable dashboard filters or drilldowns.

You can use parameters to filter widget data on a dashboard or report, and create drilldowns on dashboards. Base your filters on fields and values in the query results.

a. In the search results, identify a field by which you want to filter.

b. Using the `filter` stage, define parameters prefixed with `$`.

c. To specify parameters with a single predefined value, use the `=` operator. To specify parameters with multiple values (predefined or dynamic), use the `IN` operator.

Example of a single value parameter

Example 57. Single value parameters

The following query defines the `$domain` parameter for filtering dashboard data by domain, based on the `domain` field in the `agent_auditing` dataset.

Single value parameters are based on static predefined values. In this example, the dashboard user will be able to select a domain from a list of predefined domains.

```
dataset = agent_auditing | filter domain = $domain
```

Example of a multiple value parameter

Example 58. Multiple value parameters

The following query defines the `$endpointname` parameter for filtering dashboard data by one or more endpoint names, based on the `endpoint_name` field in the `agent_auditing` dataset.

You can configure this parameter with static predefined values, or dynamic values that are pulled from an XQL query.

```
dataset = agent_auditing | filter endpoint_name IN ($endpointname)
```

d. (Optional) Under Assign Parameters (default values), define default values for the parameters. When you add the widget to a dashboard or report, the data will be automatically populated. Alternatively, you can configure all input values when you set up a dashboard or report.

7. (Optional) Change the default time period against which to run your query from the time picker at the top right of the window. You can select the required Time frame from any of the following options available:

- Preset time ranges easily available to select from, such as 24 hours and 30 days.
- Recently used selections from your previous queries.
- Relative time: Define the time frame as the last <number> minutes, days, or hours by setting the number.
- Calendar: Create a customized time period by selecting the date range from the calendar and the specific Start Time and End Time.

**NOTE:**

- Whenever the time period is changed in the query window, the `config timeframe` is automatically set to the time period defined, but this won't be visible as part of the query. Only if you manually type in the `config timeframe` will this be seen in the query.
- These time picker options are available in XQL queries when using the Query Builder, XQL Widgets, and when defining XQL Widgets in Reports and Dashboards.

8. In the Query Results section, to graph the results either:

Use Chart Editor

Under Query Results  Chart Editor (  ), manually build and view the graph using the selected graph parameters:



- Main
  - Graph Type: Type of graphs and output options available: Area, Bubble, Column, Funnel, Gauge, Line, Map, Pie, Scatter, Single Value, or Word Cloud.

**NOTE:**

To display the result of as a time duration, choose the graph type Single Value and enable Show as Time. You can then select the Time Unit (millisecond, second, minute, or hour) and the Display format.

- Subtype and Layout: Depending on the selected type of graph, choose from the available display options.
- Header: Title your graph.
- Show Callouts: Display numeric values on graph.

- Data

- X-axis: Select a field with a string value.
- Y-axis: Select a field with a numeric value.
- (Optional) Series: For an area, bubble, column, line, map, or scatter chart, you can specify a field (column) to group chart results based on y-axis values. This option is only displayed when one of the supported graph types are selected, and a single y-axis value is selected.

- Depending on the selected type of graph, customize the Color, Font, and Legend.
- Use XQL query

Enter the visualization parameters in the XQL query section.

You can express any chart preferences in XQL. This is helpful when you want to save your chart preferences in a query and generate a chart every time that you run it. To define the parameters, either:

- Define the following query:

Example 59.

```
view graph type = column subtype = grouped header = "Test 1" xaxis = _time yaxis = _product,action_total_upload series = _vendor
```

- Select ADD TO QUERY to insert your chart preferences into the query itself.

9. Save the widget.

The custom widget appears in the list of existing widgets.

#### 3.12.4.3.1 | Configure filters and inputs for custom XQL widgets

##### Abstract

Learn more about configuring fixed filters on your dashboards to enable dashboard users to alter the scope of the dashboard.

Define fixed filters on your dashboards to enable dashboard users to alter the scope of the dashboard by selecting from predefined or dynamic values. You can define filters with free text, single select, and multiple select input values. After configuration, anyone who views your dashboard can use the fixed filters in the dashboard header.

**PREREQUISITE:**

Fixed filters are based on parameters that are defined in custom XQL widgets. Before you can configure fixed filters, take the following steps:

1. Create custom XQL widgets with parameters. For more information, see [Create custom XQL widgets](#).
2. Add the widgets to a custom dashboard. For more information, see [Build a custom dashboard](#).

##### Access and visibility

- **Role permissions:** Your ability to configure and use these filters is governed by your administrator through your user role. For more information, see [Manage access to custom dashboards](#).
- **Widget visibility:**
  - **Restricted widgets:** If you add a Restricted XQL widget with parameters to a dashboard, only users with whom you have shared that dashboard can interact with its filters.
  - **Inherited access:** When a dashboard is shared, authorized viewers can use the configured filters to narrow down the dashboard's results, even if they do not have direct access to the underlying XQL widgets in the Widget Library.

##### How to configure fixed dashboard filters

1. Initiate the dashboard editor using one of the following methods:



- From the Dashboard page: Select Dashboards & Reports â Dashboard, click the ellipse menu (vertical ellipses) in the header, and select Edit dashboard.
- From the Dashboard Manager: Select Dashboards & Reports â Dashboard Manager, right-click the custom dashboard you want to modify, and select Edit.

2. Click Add Filters & Inputs.

#### **NOTE:**

This option only appears if the dashboard contains custom XQL widgets with defined parameters.

3. Under Parameter Title, enter a name that identifies the parameter for dashboard users.

4. On the FILTERS & INPUTS panel, click +Add an input and select one of the following options:

- Single Select: To specify a single predefined value.
- Multi Select: To specify multiple predefined or dynamic values.
- Free text/number: To specify a single free text value.

#### Guidelines

- Select an option that corresponds with the parameter configured in the XQL widget query. Parameters with single predefined or free text values use the = operator, and parameters with multiple values, use the IN operator.
- Predefined values are most suitable for filtering fields that have static values, such as status fields with a limited number of available options.
- Dynamic values help you to filter with values that change often. You can configure an XQL query that extracts all of the values that are available for that field. For example, in the endpoints dataset, the endpoint\_name field values can change frequently.

5. Click Parameter and select the specific parameter from your XQL widgets that you want to configure.

#### **NOTE:**

You can define up to four parameter filters on a single dashboard or report.

6. If you selected Single Select or Multi Select values, click Dropdown Options and specify input values. When you generate the dashboard, these input values appear in a dropdown list for selection.

- To configure Predefined inputs for Single Select and Multi Select values, manually type the list values.

#### Guidelines

- The values must support the parameter type. For example, for \$name specify characters and for \$num specify numbers.
- If you uploaded numbers in a string, specify each number in quotes, for example "500".

- To configure Dynamic inputs for Multi Select values, click XQL Query to fetch dynamic values.

#### Guidelines

In the XQL Query Builder, configure a query that includes the field stage and the name of the column from which to take the dropdown values. All values in the specified field will be available for selection, and the values are dynamically updated.

#### Example 60. Example

In this example, the endpoint\_name field is configured. The dashboard user will be able to filter by one or more values from the endpoint\_name field.

```
dataset =endpoints | fields endpoint_name
```

#### **NOTE:**

If you specify more than one field, only the first field value is used.

7. Under Default Value, select a value to ensure the widget is automatically populated when the dashboard is opened.

8. Click Save Filters & Inputs and save your dashboard.

#### **TIP:**

After the initial setup, when you access your dashboard the filters and inputs might need further refinement. You can make changes to the configured parameters in the XQL widgets, and update the Filters & Inputs on your dashboard until you are satisfied with the results.



### 3.12.4.4 | Configure dashboard drilldowns

#### Abstract

Learn more about configuring drilldowns on custom dashboards providing interactive data insights when clicking data points in a widget.

#### PREREQUISITE:

To configure drilldowns your dashboard must contain custom XQL widgets. In addition, if you want to configure in-dashboard drilldowns your custom XQL widget must contain one or more parameters. For more information about configuring parameters in custom XQL widgets, see [Create custom XQL widgets](#).

Dashboard drilldowns can trigger contextual changes on the dashboard, or they can link to an XQL search, a custom URL, another dashboard, or a report. You configure drilldowns on individual widgets. After a drilldown is configured, clicking the widget triggers the drilldown. You can configure a dashboard drilldown to enable users to investigate specific data points by clicking a widget. After configuration, any user with authorized access to the dashboard can use these drilldowns.

#### Access and visibility

- **Role permissions:** Your ability to configure and interact with drilldowns is set by your administrator through your user role. For more information, see [Manage access to custom dashboards](#).
- **Inherited access:** Drilldowns respect the sharing settings of the dashboard. If you share a dashboard containing a drilldown that leads to a Restricted widget or a private query, the authorized viewers of that dashboard can still execute that drilldown and see the resulting data within the context of that dashboard.
- **Target visibility:** If a drilldown is configured to open a different dashboard, the user must have at least Viewer access to that target dashboard to view it.

#### How to configure dashboard drilldowns

1. Initiate the dashboard editor using one of the following methods:

- **From the Dashboard page:** Select Dashboards & Reports â> Dashboard, click the ellipse menu (vertical ellipses) in the header, and select Edit dashboard.
- **From the Dashboard Manager:** Select Dashboards & Reports â> Dashboard Manager, right-click the custom dashboard you want to modify, and select Edit.

2. On the widget you want to use as the trigger, click the vertical ellipses and select Add Drilldown.

3. In the Widget Drilldown dialog box, select the Action on Click:



- In-Dashboard Drilldown: Interactively filters the dashboard data. Filters are based on the parameters defined in the custom XQL widgets on the dashboard.

Define the following values:

Field	Action
Parameters	<p>Select the parameter by which to filter. You can choose any parameter that is defined in the XQL query of the widget.</p> <p><b>NOTE:</b></p> <p>If the selected parameter is configured in other XQL widgets on the dashboard, these widgets are also affected by the drilldown.</p>
Value	<p>When a user clicks the widget, the dashboard is filtered by this value.</p> <ul style="list-style-type: none"> <li>◦ Type your own value.</li> <li>◦ Select a variable from which to capture the clicked value, for example, the \$y-axis.value in a chart. For more information, see Variables in drilldowns.</li> </ul>

- Link to dashboard: Opens a target dashboard.

Define the following values:

Field	Action
Dashboard	Select the target dashboard.
(Optional) Parameter	Select parameters by which to filter the data on the target dashboard. Parameters are only available if there are parameters defined in the widgets on the <i>target</i> dashboard.
(Optional) Value	<p>When a user clicks the widget, this value is configured as a parameter on the target dashboard.</p> <ul style="list-style-type: none"> <li>◦ Type your own value.</li> <li>◦ Select a variable from which to capture the clicked value in the source dashboard, for example, the \$y-axis.value in a chart. For more information, see Variables in drilldowns.</li> </ul>

- Open XQL Search: Runs an XQL query based on the clicked value.

Define the following values:

Field	Action
XQL Query	<p>Define the query that you want to run on drilldown.</p> <p>Type \$ to see autocomplete options for variables that are available in the widget drilldown. For example, in a table widget \$first.name selects the leftmost column name in the table. For more information, see Variables in drilldowns.</p>

In the following example two parameters are passed from a table widget to an XQL query. The first parameter with the cell value that the user clicked on, and a second parameter with the cell value in the request\_url column in the row that the user clicked.

```
dataset=xdr_data
|filter event_type=$y_axis.value and requestUri=$row.request_url
|fields action_download, action_remote_ip as remote_ip,
actor_process_image_name as process_name
|comp count_distinct(action_download) as total_download by process_name,
remote_ip, remote_hostname
|sort desc total_download
|limit 10
|view graph type=single subtype=standard xaxis=remote_ip yaxis=total_download
```



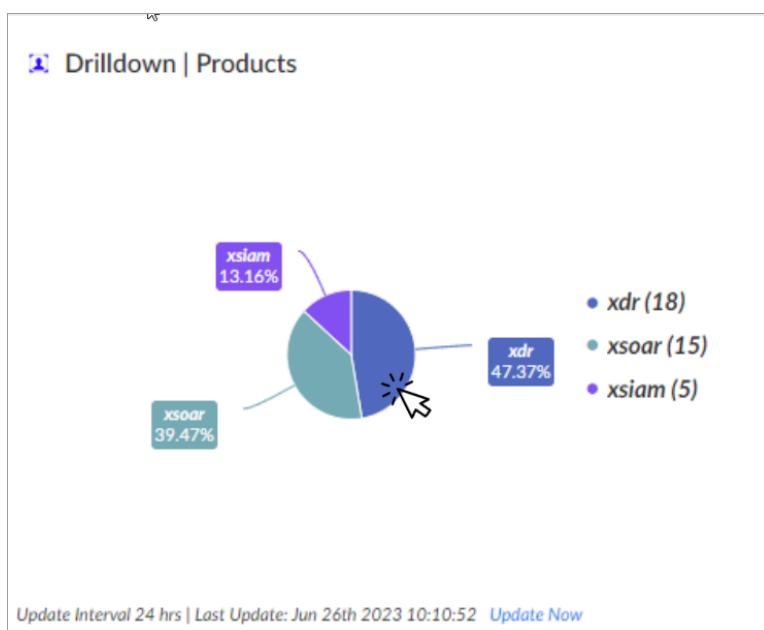
- Open custom URL: Opens an external URL based on a clicked value.

Define the following values:

Field	Action
URL Address	Type the URL. To create a dynamic drilldown, you can include Available parameters. For more information about the parameters, see Variables in drilldowns.

In the following URL, the `$x_axis.value` parameter represents cortex products names. On drilldown, the `$x_axis.value` is replaced with the clicked product name in the pie chart.

[https://www.paloaltonetworks.com/cortex/cortex-\\$x\\_axis.value](https://www.paloaltonetworks.com/cortex/cortex-$x_axis.value)



- Generate Report: Runs a report from a clicked value.

4. Save the changes for the widget and dashboard.

3.12.4.4.1 | [Variables in drilldowns](#)

#### Abstract

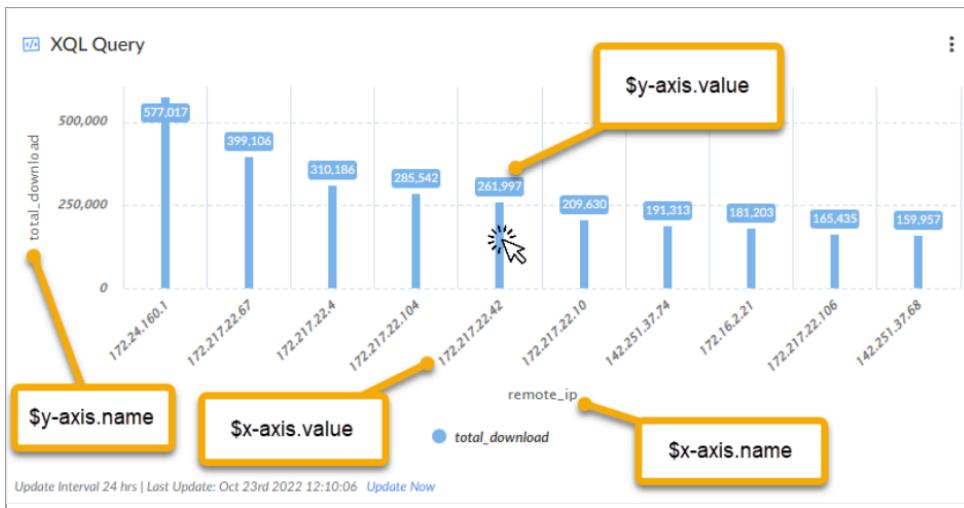
Learn about the widget variable values that you can use in dashboard drilldowns.

The following tabs are organized according to widget type and describes the widget variables that are available in drilldowns. The variable defines the value to capture in the drilldown, according to the element that is clicked. The captured value is then configured as a parameter by which to filter data on drilldown.

#### Chart

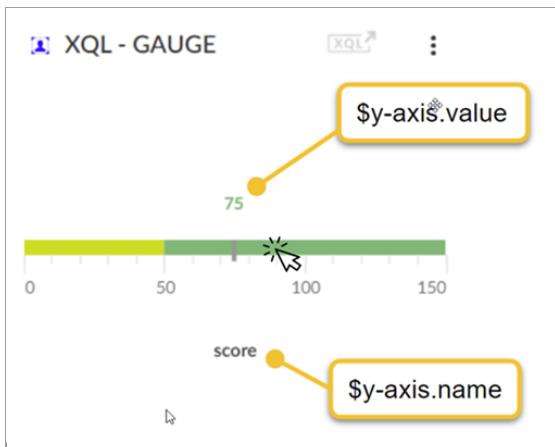
(Area, Bubble, Column, Funnel, Line, Map, Pie, Scatter, or Word Cloud)





- **\$x\_axis.name**: Selects the x-axis name.
- **\$x\_axis.value**: Selects the x-axis value for the clicked value.
- **\$y\_axis.name**: Selects the y-axis name.
- **\$y\_axis.value**: Selects the y-axis value for the clicked value.

Single value or gauge



- **\$y\_axis.name**: Selects the y-axis name that the single value represents.
- **\$y\_axis.value**: Selects the y-axis value for the clicked value.

Table

**Table Widget**

The table has the following data:

_TIME	ACTION_TOTAL_UPLOAD	_VENDOR	INSERT_TIMESTAMP	_PRODUCT
Oct 22nd 2022 14:35:28	13008	I	Oct 23rd 2022 07:26:12	Fusion
Oct 22nd 2022 14:25:48	9922	PANW	Oct 23rd 2022 07:31:40	Fusion
Oct 22nd 2022 14:35:28	13148	PANW	Oct 23rd 2022 07:21:40	Fusion
Oct 22nd 2022		PANW	Oct 23rd 2022 07:52:52	Fusion
Oct 23rd 2022 02:12:43	32	PANW	Oct 23rd 2022 04:06:15	Fusion

Update Interval 24 hrs | Last Update: Oct 23rd 2022 11:57:25 [Update Now](#)



- `$first.name`: Selects the leftmost column name in the table.
- `$first.value`: Selects the leftmost value in the clicked table row.
- `$clicked.name`: Selects the column name of the clicked value.
- `$clicked.value`: Selects the value in the clicked table cell.
- `$row.<field_name>`: Selects the field (column) from the clicked table row.

## 4 | Investigation and Response

### Abstract

Learn how to investigate cases and issues, run commands, use XQL to run queries, and use the Cortex Copilot.

Investigate cases and issues, run commands, automations, and XQL queries.

### 4.1 | Overview of cases

Understand how cases work in Cortex AgentiX.

#### 4.1.1 | What are cases?

##### Abstract

A case provides the full contextual story of a problem that impacts your organization's security, giving you an end-to-end view of the problem and streamlining your understanding of what needs to be solved and how.

A case is a defined problem created by connecting related issues into a single story. It shows the impacted assets and key data in one place, helping you focus on the threats that matter most, reduce noise, and resolve the problem efficiently using automation. Each case is unique and requires its own investigation.

Cases comprise the following objects:

- **Issues**: Problems detected in your environment that exceed defined thresholds or surpass your organization's accepted level of risk and threat tolerance.
- **Assets**: Specific entities impacted in a case and how they fit into the case story.
- **Artifacts**: Objects to which behavior or influence can be attributed, such as filenames, processes, domains, and IP addresses.

To see a list of all cases, go to Cases & Issues → Cases.

While cases are configured to work OOTB, users with specific requirements can customize and tailor their cases. For more information, see Customize cases and issues.

##### Case creation

A case can be created automatically from an issue or manually by a user. When new issues are detected, Cortex AgentiX checks them against existing cases. If there is no matching case, a new case is created. When an issue is linked to a case, all associated assets and artifacts are also linked. After case creation, new issues can match the case until the grouping threshold is met.

A case is automatically generated for any issue with Medium severity or higher that falls into one of these categories:

- It is assigned to the Security domain.
- It was generated from the public API or created from correlations.

While most low-severity issues do not create cases, specific analytic rules can trigger case creation for low-severity issues when action is deemed necessary. Low-severity issues created from correlation rules are not grouped into cases.

For more information about how cases are built, see Case grouping.

#### 4.1.2 | Resolving cases with AI

##### Abstract

AI tools can help you through the case analysis and resolution process.

To simplify and accelerate case resolution, Cortex AgentiX integrates advanced generative intelligence directly into the case management lifecycle. By leveraging built-in machine learning and intelligent grouping logic, Cortex AgentiX shifts the focus from resolving isolated issues to a holistic approach that resolves the case as a whole:



- Intelligent case grouping:** Cortex AgentiX automatically consolidates related issues, assets and artifacts into a single unified case that reveals the full scope of an attack.
- SmartScore prioritization:** Each case is assigned a SmartScore based on its severity and calculated risk. This enables teams to focus on the most critical cases first, ensuring that high-impact security threats, posture gaps, or health issues are handled with appropriate urgency.
- AI summarization:** Agentic AI is integrated in the case resolution process to automatically summarize context, help you investigate entities, and suggest remediation actions.
- Guided resolution:** The Resolution Center guides you to resolution with actionable tasks that are designed to remediate the entire case as a single entity, significantly accelerating the path to resolution.

#### Agentic AI

Cortex AgentiX leverages Agentic AI to collaborate on investigations and actively accelerate the entire resolution lifecycle.

Feature	Description
AI-generated case summaries	Instantly analyzes the case's full scope and impact and accelerates triage.
Agentic Assistant	The autonomous "brain" of Cortex AgentiX. It utilizes AI agents that plan, reason, and investigate complex threats, such as cloud identity theft or container breaches. These agents have access to case context and can create plans and perform actions such as running commands, playbooks, and scripts.  The Agentic Assistant chat provides an interactive and intelligent way to simplify and streamline complex security operations. Enter a prompt using natural language, and your agent plans and executes the most relevant actions to fulfill your request.
Resolution Center	Provides actionable remediation tasks, recommendations, and progress tracking to guide you step-by-step to a complete resolution.  With playbook task tracking across all issues and in-context links to the Workplan, you can manage tasks awaiting action, monitor work in progress, and review completed items.

#### 4.1.3 | Case lifecycle

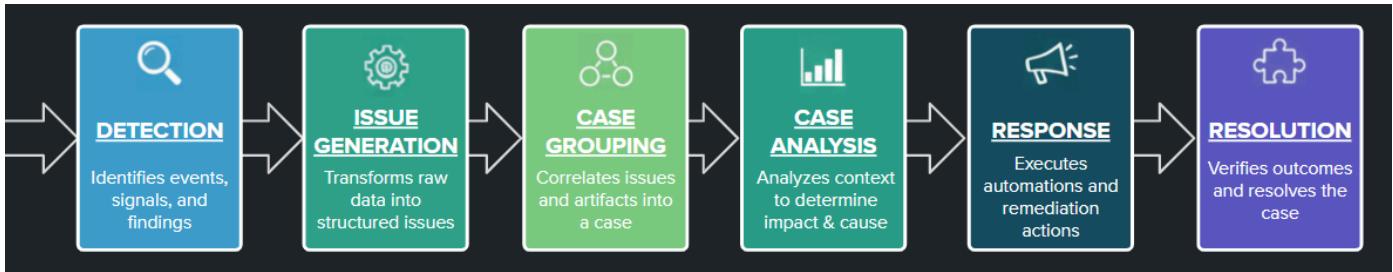
##### Abstract

Understand the lifecycle of a case.

Cortex AgentiX handles cases through a structured process that moves from identification to resolution.

Stage	Description
Detection	Signals or findings surface across the environment.
Issue generation	Raw data is converted into structured, defined as Issues.
Case grouping	Issues are evaluated for case qualification. If the issue qualifies it is grouped into a case with related issues, or if no match is found, a new case is generated.
Case analysis	Examination of context, relationships, and evidence.
Response	Application of remediation actions to mitigate the threat.
Resolution	Final confirmation that the issues in the case are fully addressed.





#### 4.1.4 | Case thresholds

##### Abstract

Case grouping thresholds are implemented to keep cases manageable.

To keep cases manageable, Cortex AgentiX implements case grouping thresholds. When the case reaches a threshold, it stops accepting issues and groups subsequent related issues in a new case.

- 30 days have passed since case creation.
- 14 days have passed since the last issue was detected.
- A case reaches the 1,000 issue limit.

You can track the threshold status in the `Issues Grouping Status` field in the cases table.

##### Auto-resolved cases

If a case is resolved with the status `Resolved - Auto Resolved`, Cortex AgentiX reopens the case within a six-hour window if a matching issue occurs. The six-hour period is defined by the timestamp of the last issue that was grouped into the case. After the six-hour period, any new issues are linked to a new case for a new investigation.

#### 4.1.5 | Case scope and impact

##### Abstract

A case's scope and impact is determined by the assigned severity, score, and domain.

The prioritization and governance of cases are determined by the case `Severity`, `Score`, and `Domain`. Together, these factors define the operational urgency and the investigative boundaries of a case.

- **Severity:** This attribute reflects the immediate risk level. Cortex AgentiX employs a logic where the overall case severity is dictated by the most critical issue linked to it. This ensures that high-impact threats are instantly visible to responders without being diluted by lower-level activity.
- **Score:** The case score provides a quantitative measure of risk. While severity indicates the severity of a case, the score offers a granular numerical value used for precise ranking.
- **Domain:** This categorizes the case context for example Security or Health. The domain determines the case's scope, directing it to the appropriate specialized team.

By aligning these three factors, Cortex AgentiX automates the transition from detection to response, ensuring the most critical risks are addressed by the right experts.

#### 4.1.6 | Case and issue domains

##### Abstract

Cortex AgentiX assigns each case and issue to a domain. Domains help you to organize and manage your work efforts, and differentiate between use cases.

Depending on the objects identified in a case or issue, each case and issue is assigned to a domain that reflects the root cause and the system areas of operation.

Domains are a contextual boundary that allow you to manage and prioritize each use case and help you to differentiate between your security use cases and non-security use cases. Domains help you to organize and manage your work efforts, streamline the assignment of cases, and enable you to create tailored experiences for each domain.

When an issue is created, Cortex AgentiX automatically assigns it to a domain, and the same domain is assigned to the associated case. If you create your own case, you can select the domain to which you want to assign it.



Each case and issue is assigned to a single domain. You cannot change the assigned domain, however cases can be linked to issues from different domains.

#### Built-in domains

Cortex AgentiX provides the following built-in domains:

Domain	Description
Security	<p>For cases and issues that are associated with case response activities for detecting, preventing, and blocking threats as they occur in runtime.</p> <p>For example, the identification of malware in a file, a compromised endpoint, or a phishing attempt. These cases can be assigned to a SOC analyst who specializes in blocking and remediating attacks.</p>
Health	<p>For cases and issues that are associated with health monitoring activities, to ensure optimal platform performance and gain insights into health drifts. For example, disruptions in data ingestion, collector connectivity errors, correlation rule errors, and event forwarding errors.</p>

## 4.2 | Case concepts

### 4.2.1 | Issues, findings, and events

#### Abstract

Understand how issues, findings, and events are related to cases.

Understand how issues, findings, and events are related to cases.

#### 4.2.1.1 | Issues

Issues identify the problems that you need to solve in your environment. Cortex AgentiX creates issues when problems occur in your environment that cross defined thresholds, or surpass your organization's accepted level of risk and threat tolerance.

Each issue comprises a defined framework of:

- **What happened:** A description of the problem
- **How is your environment impacted:** Affected assets or the impact of this issue in your environment
- **Contributing evidence:** Data that supports our analysis and observations
- **Recommended actions:** Automations, Playbooks, and manual suggestions

Issues are created from findings or from events that occur in your environment. When an issue is created, Cortex AgentiX assesses the content of the issue and assigns it to a new or existing case. In addition, according to the content of the issue, it is assigned to a domain that reflects the operational use case of the issue, such as Security or Health. Using case grouping logic, Cortex AgentiX then determines whether to link the issue to a case.

When you open a case, you can see all issues that are linked to the case. Review the **Grouping graph** to see why the issues were grouped together in the case. For more information about how issues are grouped in cases, see [Case grouping](#).

In addition, Cortex AgentiX offers the flexibility to:

- Mirror Cortex issues with external applications (for example, Atlassian Jira). For more information, see [Issue syncing](#).
- Create issues from custom rules that you define. For example, correlation rules. For more information about setting up rules, see [What's a correlation rule?](#).

#### 4.2.1.2 | Findings and events

#### Abstract

Findings and events form the core of our knowledge data lake. **Findings** provide context about the current state of the assets in your environment and **Events** are logged activities that occur in your environment.

Findings and events form the core of our knowledge data lake.



## Findings

**Findings** are non-actionable, informational objects that provide context about the *current state* of the assets in your environment.

To gather findings, Cortex AgentIX periodically scans the assets in your environment and collects raw data about vulnerabilities, compliance, exposures, malware, secrets, and other posture-related information about the asset. This raw data is processed, saved to datasets, and recorded as findings.

Each time the assets are scanned, the findings are updated to reflect the current state of the assets. Therefore, the finding for an asset will change over time.

Each finding is categorized according to its context, for example Configuration, Vulnerability, Compliance, or Identity, and is related directly to the scanned asset. When you investigate an asset through the Asset Inventory, you can see any findings that were collected for the asset.

Findings themselves are not issues, however findings that match a specific logic can generate issues. You can also set up your own rules to trigger issues when certain types of findings are recorded. For example, you can set up Compliance rules that will create issues if specific compliance fails are identified in compliance findings.

To view findings:

- View all findings. From the Issues page click Findings.
- See findings for a specific asset. From the Asset Inventory, select a specific asset to open the asset card. If findings are available for the asset you can click to open the finding card.
- Search the **Findings** data set to see the findings collected over time for an asset.

## Events

**Events** are logged activities that occur in your environment.

Cortex AgentIX collects event logs that audit the activities that occur in your environment. The logs are ingested from various sources, such as Palo Alto Networks Next-Generation Firewall (NGFW), Prisma Access, third-party sources, and EDRs. These logs provide a complete picture of the events that occur in the environment and the activities surrounding the events.

When certain malicious objects (such as malware) are discovered in the event logs, an issue is created. During case investigation, you can query your event logs to see information about the actors and processes that triggered the issue.

### 4.2.2 | Case grouping

#### Abstract

Cortex AgentIX uses a specific case grouping logic to build cases.

Case grouping is a Precision AI-powered capability that eliminates alert fatigue by automatically consolidating related issues and artifacts into a single unified case. Case grouping links issues that originate from the same attack flow or involve the same entity to reveal the full scope of a case. This approach replaces manual correlation with automated context, allowing you to focus on resolving complete problems rather than triaging isolated events.

#### Grouping methodologies

The key grouping methodologies of case grouping are:

- **Artifact association:** Groups issues that share core artifacts (for example, SHA256, HostName, UserName).
- **Exact match detection:** Groups similar detections for the same entities.
- **Related entities:** Groups detections involving related assets within a close timeframe to highlight possible connections.

#### Case qualification for issues

Not all issues create cases. When a new issue is created, it is evaluated to determine if it meets the criteria for case promotion. If the issue qualifies, the system attempts to correlate it with an existing case; if no match is found, a new case is generated. Issues that do not meet these requirements are categorized as Insights.

The qualification logic varies by domain. For the Security domain, the system promotes issues with Medium severity and above, as well as select Low-severity analytics. Other domains employ more selective promotion based on specific criteria. This logic is dynamic and may be updated to reflect ongoing research and threat relevance.

Cortex AgentIX applies the following logic when building cases:

- **Automatic promotion criteria:** Issues with the following conditions automatically generate a new case, or join existing cases:
  - Assigned to the **Security** domain with **Medium** severity or higher
  - Generated from the **public API** or created from **correlations**.



- **Low severity handling:** Most low severity issues do not initiate case creation, unless specific analytic rules deem action necessary. Low severity issues generated from correlation rules are not grouped into cases.
- **Case grouping thresholds:** To keep cases manageable, Cortex AgentiX enforces specific grouping thresholds. For more information see Case thresholds.

#### Grouping artifacts

The grouping algorithm evaluates extracted artifacts to determine whether an issue should join an existing case or initiate a new one. Each artifact type is governed by specific logic that accounts for its unique lifecycle and reliability. For example, grouping by Username may be subject to temporal constraints, while IP address logic varies based on whether the address is public, private, or dynamically allocated (DHCP).

These proprietary grouping logics are continuously tuned and updated. As a result, artifact behavior and correlation may change over time.

#### Limitations

Case grouping is natively supported within built-in domains only, for example Security.

### 4.2.3 | Case scoring

#### Abstract

Learn about the different case scoring methods.

A case score is a numeric value that indicates the urgency of a case. Scoring can help you to streamline the process of prioritizing and investigating your cases, and help you to identify the cases that require immediate attention.

#### Types of scoring

Cortex AgentiX uses the following scoring methods:

- Rule-based scoring: The score is determined by user-defined scoring rules that match the issues linked to the case.

You create scoring rules that define scores for issues with specific attributes or assets. You can base scoring rules on:

- Hostnames
- Asset objects, such as asset names, classes, categories, groups, providers, and business application names.
- IP addresses
- Users
- Active Directory, or Azure groups and organization units

(Requires the Cloud Identity Engine to be configured).

When an issue is created, Cortex AgentiX searches for scoring rules that match the issue. An issue can match multiple rules or sub-rules. If a match is found, Cortex AgentiX assigns the scores of the matching rules to the issue. If multiple rules match the issue, the issue score is an aggregation of the rule scores. By default, a score is applied only to the first issue in the case that matches the defined rule and sub-rule.

You can create a rule hierarchy by setting up sub-rules. If an issue matches one or more sub-rules, the sub-rule scores are also aggregated in the issue score. However, a sub-rule score is only applied to an issue if the top-level rule was a match.

To determine the case score, Cortex AgentiX calculates the combined issue score total for all issues in the case. You can see a breakdown of the score by clicking on the score in the details pane.

- Manual scoring: The score is defined by the user.

#### How Cortex AgentiX assigns the score

For Cortex AgentiX to provide effective rule-based scores, you must define accurate scoring rules that are suitable for your environment and workflows.

When a case is created, Cortex AgentiX searches for a match between your scoring rules and the issues linked to a case. If a match is found, a rule-based score is assigned.

You can view the assigned score on the Cases page.

### 4.2.4 | Case starring

#### Abstract

Starring cases can help you to prioritize and filter your cases.



To help you focus on the most important cases, you can star a case. Starring enables you to narrow down the scope of cases on the Cases page. Cortex AgentiX identifies starred cases with a purple star.

You can star cases manually, or create a starring configuration. A starring configuration automatically categorizes and stars cases that contain issues with specific attributes. For example, you can define a starring configuration that stars all issues containing specific assets, hosts, or business application names. If an issue matches the attributes in the starring configuration, the issue and case linked to the issue are starred.

You can manage all starring configurations under Case & Issues → Case Configuration → Starred Issues. For more information see Create a starring configuration.

#### 4.2.5 | SLAs and tracking

Abstract

You can set up Service Level Agreements (SLAs) to track your cases against SLA targets.

In Cortex AgentiX, Service Level Agreements (SLAs) are tracked using a combination of Timer fields and SLA fields. This system allows you to quantify performance and ensure that critical security cases are addressed within defined timeframes.

SLA configuration components

- **Timers:** These fields count forward to measure the actual duration of an action. For example, a **Time to Assignment** timer starts when a case is created and stops when an owner is assigned.
- **SLA fields:** These fields count backward from a predefined goal. They visualize the time remaining until a deadline is breached, changing color, for example to red, if the goal is exceeded.
- **Severity-based goals:** You can define different SLA targets based on case severity. This ensures that Critical cases receive a faster response than Medium or Low severity cases.

For more information about configuring SLAs and timer fields, see Create case timers and SLAs.

### 4.3 | Analyze and resolve cases

Abstract

Learn how to analyze and resolve cases.

The following sections explain how to review, analyze, and resolve cases. You can start reviewing the cases in your environment on the Cases page.

#### 4.3.1 | Review all cases

Abstract

Start reviewing your open cases on the **Cases** page.

The main **Cases** page is the starting point for monitoring and managing all cases in your environment. It provides visibility into all cases and their current status, helping you track progress, investigate individual cases, and take remediation actions. Severity indicators, scores, and starred icons help you quickly identify your high-priority cases.

When cases are configured with SLAs, the page helps you monitor SLA adherence and ensure cases progress in line with organizational objectives.

You can access the **Cases** page from **Cases & Issues** → **Cases**. By default, all open cases are displayed.

Viewing modes

The cases page supports the following viewing modes:

- **Split view (default)**

Displays cases in a split-pane layout that highlights key details and enables you to quickly compare cases, prioritize urgent items, and assess severity and impact at a glance.

- **Table view**

Displays cases in a table layout with widgets that summarize the table data. Widgets are customizable, allowing you to tailor the table for structured analysis and review.

Click the **Display** menu to switch between modes. Any changes that you make to the case fields persist between modes.

#### NOTE:

The legacy view is also available for users who prefer this format. From the Actions menu select **Switch to legacy view**.



## Saved table views

Saved table views are saved filter configurations of table data that help you to focus on the data that most matters to you. You can filter your table data by domain, context, work queue, or other criteria, and save configurations that support your workflow.

The default view on the Cases page is **All Cases**. Click on the arrow next to All Cases to see all available saved views. If you change the table filters, you will see a Modified label next to the view name. You can create a new saved views. Once you have change the table filters, click the three dots next to the view name to save the new configuration, update an existing saved view, or revert to the original configuration.

### 4.3.2 | Start case analysis

#### Abstract

Understand the case analysis and resolution process.

To start analyzing a case, open the case from the main **Cases** page. In the Split view, click a case to open it in the side panel. To open a case in a full page layout, right-click a case in the list and select View case in new tab.

The case card opens a dedicated workspace where you can fully understand, investigate, and resolve the case from start to finish.

The case card brings together case context, correlated issues, affected assets, and remediation actions in one place. It helps you quickly understand the case context, see how events are connected, and take action with confidence. Click through the view to dive into investigation data, resolution tasks, and AI assistance without switching pages or losing context, keeping your focus on resolution.

#### Case analysis and resolution process



#### Core components

The following table describes the core components of case analysis and resolution:

Component	Description	Link To Detailed Information
Agentic Assistant	Provides side-by-side support by recognizing case context, delivering advanced summarization, and helping you pivot to additional investigative views.	Agentic Assistant- Case Investigation agent
AI-generated case title and description	Helps you quickly understand the scope and nature of the case by summarizing key case details.	AI-generated case summaries



Component	Description	Link To Detailed Information
Case overview	<p>Breaks down case components to help you understand how the case was built:</p> <ul style="list-style-type: none"> <li>• <b>Grouping graph:</b> Illustrates issue relationships</li> <li>• <b>Evidence:</b> Details casualties and events</li> <li>• <b>Issue feed:</b> Narrates the case story</li> <li>• <b>Associated assets, artifacts, and MITRE ATT&amp;CK tactics:</b> Provides additional context and links to detailed views and actions</li> </ul>	Analyze case details
Detailed view	Provides detailed information about the investigation in a tabular format, for example Timeline and War Room.	Detailed View
Resolution Center	Guides you towards resolution by presenting actionable remediation steps and enables you to track all related playbook tasks without opening individual playbooks.	Resolution Center

#### 4.3.2.1 | Agentic Assistant- Case Investigation agent

##### Abstract

The Agentic assistant provides side-by-side support throughout the case analysis and resolution process.

The **Agentic Assistant** is a context-aware, generative intelligence tool embedded directly within the case card. It is designed to act as a side-by-side partner for security analysts, eliminating the need to pivot away from the investigation to consolidate complex data.

When you open the Agentic Assistant you can select the agent that is best suited for each task. The dedicated Case Investigation agent can help you with your case investigation. It specializes in advanced summarization, and recognizes the context of the case, ensuring every insight provided is highly relevant and grounded in the specific issues, assets, and telemetry of the current investigation.

For more information about using other agents in the Agentic Assistant, see Get started with Agentic Assistant chat.

##### Core functionalities of the Case Investigation agent

To streamline case analysis, the assistant provides the following areas of support:

- Dynamic summarization of log data and issues into clear, actionable narratives, including:
  - **Executive overviews:** High-level summaries that focus on impact and risk.
  - **Extended technical overviews:** Deep-dive summaries that outline the technical progression of the threat.
- Focused contextual inquiries to extract specific details without manual filtering. You can ask targeted questions regarding:
  - **Issue deep-dives:** Understanding the specific triggers and severity of an issue.
  - **Asset relationships:** Identifying which users or devices are at the center of the activity.
  - **Asset and artifact investigation:** Understanding the impact and risk of the assets and artifacts in the investigation.
- Intelligent pivoting and clarification to help you navigate through complex investigations:
  - **Entity-specific prompts:** By clicking **Ask AI** next to a specific entity (such as an IP address or file hash), the assistant launches with a pre-configured prompt tailored to that specific object.
  - **Investigation guidance:** It suggests potential next steps and actions, and links to detailed views

#### 4.3.3 | Establish case context

Before you start to analyze the case, review the case title and description to establish case context. You can also review the case score, assignee, and decide whether to star the case.



#### 4.3.3.1 | AI-generated case summaries

##### Abstract

AI generated case summaries helps you quickly understand the scope and nature of the case by summarizing key case details.

To gain immediate situational awareness, Cortex AgentiX automatically builds a narrative of the case using **AI-generated titles and descriptions**. This summarized context allows you to quickly grasp the scope of a case and provides a clear starting point for your investigation.

Leveraging LLM-based summarization, the system analyzes complex data to produce a human-readable overview of:

- The nature of the threat or activity
- The key issues and artifacts involved
- The affected assets or identities

[View the AI-generated case summary](#)

When you open a case, the case title and summary is automatically generated. As an investigation evolves, the case context is updated. Each time new data or issues are added, the system regenerates the title and description to ensure your situational awareness reflects the most current information available.

##### NOTE:

The AI-generated title and description is a calculated value that is regenerated each time you open a case.

This value is not a saved static description, therefore it is not reflected in the saved case names in the list of cases in the **Split view**, or in the **Case Name** and **Case Description** columns in the **Table view**.

[System-generated case titles and descriptions](#)

In addition to the AI-generated case titles and summaries, Cortex AgentiX automatically generates static case titles and descriptions that are stored in the cases dataset. These are generated at the time of case creation based on correlated issues, behaviors, and contextual data.

These static descriptions are used when AI-generated case summaries are unavailable or disabled. In addition, they are reflected in the case title in the List of cases in the **Split view**, and the **Case Name** and **Case Description** columns in the **Table view**.

You can manually update these values. From the **Actions**  menu select **Edit case details**.

[Single issue cases](#)

For cases that contain a single issue, the case title and description directly reflect the issue's title and description. In addition, AI-generated case summaries are not available. If more issues are linked to the case, Cortex AgentiX generates a case title and description to reflect the issues in the case, and a AI case title and summary is available.

[Limitations](#)

- **Supported regions:** AI-generated case titles and summaries are available only in supported regions. For more information, see [Cortex AgentiX Assistant](#).
- **Supported domains:** AI-generated case titles and summaries are only supported for cases assigned to the **Security** domain.
- **Single-issue cases:** For cases that contain a single issue, AI-generated case summaries are not available. Instead, the case title and description directly reflect the issue's title and description. If more issues are linked to the case, an AI case title and summary is generated.

[Enable AI summarization](#)

To enable AI case summarization on your tenant, go to **Configurations**  $\rightarrow$  **General**  $\rightarrow$  **Server Settings**  $\rightarrow$  **AI Configuration** and enable the following settings:

- Agents & LLM Experience
- AI Case Summarization

You can also turn AI summarization on or off for a specific case. Take the following steps:

1. Open the case, click the **Actions** menu.
2. Select **Edit case details**.
3. Switch the **Summarize with AI** toggle.

#### 4.3.3.2 | Assess case severity and score

##### Abstract



Review the case severity and score, and see a breakdown of how the score was calculated.

You can review the severity and score assigned to the case, and update them if necessary.

#### Review case severity

The severity value indicates the urgency of a case. Possible values are **Critical**, **High**, **Medium**, and **Low**. Click on the assigned severity to change the value.

#### Review the case score

The assigned case score is displayed in the cases header. This score indicates the urgency and impact of the case.

Click on the case score to see the assigned scoring method. For more information about scoring types and how Cortex AgentiX assigns a score, see Case scoring.

#### See a breakdown of the score

You can see details about the scoring method and the assigned score.

1. On the **Cases** page, click on the menu icon to switch to the detailed view.
2. Click on an assigned score.

If you are not satisfied with the score, you can change the scoring method or overwrite the score by setting the score manually. If you see a discrepancy with the assigned score, consider the following:

- For rule-based scores, revise your scoring rules.

#### Change the scoring method or set the score manually

You can change the default scoring method. In addition, if Cortex AgentiX was unable to assign a score, you can set the score manually.

1. Click on the assigned score.  
If no score was assigned, in the case investigation pane, click the more options icon and select **Manage Score**.
2. Select a different scoring method, or click **Set score manually** and define a new score.

### 4.3.3.3 | Update case attributes

#### Abstract

You can update the case title and description, and choose whether to star a case.

When you start reviewing a case, you can update the case title and description, assign the case, and star a case.

#### Assign a case

You can assign or reassign a case by clicking on the assigned field.

If the case contains unassigned issues, or the issues are not assigned to the case assignee, a dialog opens with options for assigning the issues.

#### Update the case title and description

A case title and description is automatically generated for each case. In addition, AI-generated case summaries are automatically generated when you open a case to provide case context.

You can manually update the saved case description, as required.

1. Select a case and open the Actions  menu.
2. Select **Edit case details**.
3. Update the values in the **Case title** and **Case description** fields.

#### NOTE:

The defined values are shown in the **Case Name** and **Case Description** columns in the Table view, and saved to the **cases** dataset. The case title is also shown in the list of cases in the Split view.

These values do not replace the AI-generated case title and summary. If the **Summarize with AI** toggle is enabled, AI-generated case summaries are automatically generated when you open a case. For more information about how Cortex AgentiX generates case titles and descriptions, see AI-generated case summaries.

4. Save your changes.



## Star or un-star a case

You can manually star or un-star a case:

1. Go to Cases & Issues â Cases and select the case that you want to star.
2. Depending on the selected view, take the following action:
  - In the **Split** view, open the Actions menu and select Edit case details. Switch the toggle to star or un-star the case.
  - In the **Table** view, select one or more cases and right-click. Select whether to star or un-star the cases.

### 4.3.4 | Analyze case details

#### Abstract

You can analyze detailed information about the case in the **Overview** section of the Case card.

Once you have established the initial context, you can use the case **Overview** to deconstruct the case and understand how its underlying components are connected. Use the following sections within the **Overview** to review the full scope of activity:

- **Grouping Graph:** View a visual mapping of how issues and artifacts are linked together, including details on shared artifacts, to better understand the underlying grouping logic.
- **Evidence:** Trace issue causality chains and recorded events to follow the attack sequence from the initial root cause to the final recorded activity.
- **Issue feed** Review the caseâ€ s story in a chronological visualization that maps the case lifecycle and highlights key case information, with the option to group by attribute.
- **Associated assets and artifacts:** Drill down into the specific identities, endpoints, and digital artifacts associated with the case to assess the threat's footprint.
- **MITRE ATT&CK tactics and techniques:** Review the specific tactics and techniques identified in issues linked to the case to align your investigation with industry-standard adversary behaviors.

The following topics describe each section of the Case **Overview**.

#### NOTE:

If you prefer a tabular or legacy layout, switch the case card to the **Detailed view**.

This view preserves the legacy tab based format and custom layouts, ensuring full backward compatibility. You can switch between the new case experience and the legacy view based on personal workflow preferences. For more information, see [Detailed View](#).

#### 4.3.4.1 | Grouping graph

#### Abstract

Gain insight into why issues were grouped in a case.

The **Grouping Graph** is a visual representation of the logic used to group issues in a case. It provides transparency into why specific issues are linked, illustrating the relationships between data points and the underlying decision-making process of the analysis engine.

By revealing these connections, the graph offers key insights into the case narrative, visualizes the overall scope, and identifies common artifacts for investigation.

#### Understanding case grouping

Cortex AgentIX automatically matches issues and artifacts into a unified case based on a specific grouping logic. This allows you to resolve the entire scope of a case rather than treating detections in isolation. The logic is driven by the following factors:

- **Artifact association:** Issues sharing core artifacts, for example the same file hash or IP.
- **Similarity clustering:** Issues with similar detection patterns on the same entities.
- **Related entities:** Detections on related assets occurring within a close timeframe or context.
- **Linked and merged issues:** Issues that were manually linked to the case and merged issues.

Related issues are added to the case until a specific **grouping threshold** is met. In the **Grouping Graph** you can see whether case grouping is active or inactive. For more information about case grouping and case thresholds, see [Case grouping](#).

#### Core components of the Grouping Graph

The graph uses a structured hierarchy of edges and nodes to represent the primary elements of a case:



Component	Description
Edges	<p>Represent the relationship between graph entities to show why they were linked. Edges display as lines that link nodes and entities together. Each full line represents a direct relationship.</p> <p>The system defines three edge types:</p> <ul style="list-style-type: none"> <li>• <b>Case &gt; Issue:</b> Links the case to the issue that initiated its creation.</li> <li>• <b>Issue &gt; Artifact:</b> Links an issue to an associated artifact. This indicates that the issue is the source of the artifact in the case.</li> <li>• <b>Artifact &gt; Issue:</b> Links an artifact to an issue or issue cluster. This indicates that the artifact is the source of the issues in the case.</li> </ul> <p>Edges display as:</p> <ul style="list-style-type: none"> <li>• <b>Solid line:</b> Connects the case node to its originating issue, as well as to related artifacts and additional issues later grouped into the case.</li> <li>• <b>Broken line:</b> Connects similar, manually linked, or merged issues to the case. The connection type is indicated by a label: <ul style="list-style-type: none"> <li>◦ <b>linked:</b> Issues manually linked to the case</li> <li>◦ <b>similar:</b> Issues grouped by similarity clustering</li> <li>◦ <b>merged:</b> Issues merged into the case</li> </ul> </li> </ul>
Case node	The central anchor node to which all other elements are connected.
Issue nodes	Visualized with parent/child relationships to show how primary threats spawned secondary activities.
Clusters	<p>Groups of issues that are automatically clustered to keep the visual workspace organized, with details of the total issue count in the cluster and severity breakdown. Issues are clustered if they:</p> <ul style="list-style-type: none"> <li>• Share a common artifact.</li> <li>• Are manually linked to the case.</li> <li>• Have been merged.</li> <li>• Are identified as similar through similarity clustering.</li> </ul> <p><b>NOTE:</b></p> <p>Similar issues are displayed as individual entities rather than in a parent/child hierarchy.</p>
Artifacts	Represent artifacts that are linked to the issues in the case. Artifacts include user names, IPs, and causality chains. Causality chains link issues in the same causality chain to the case.

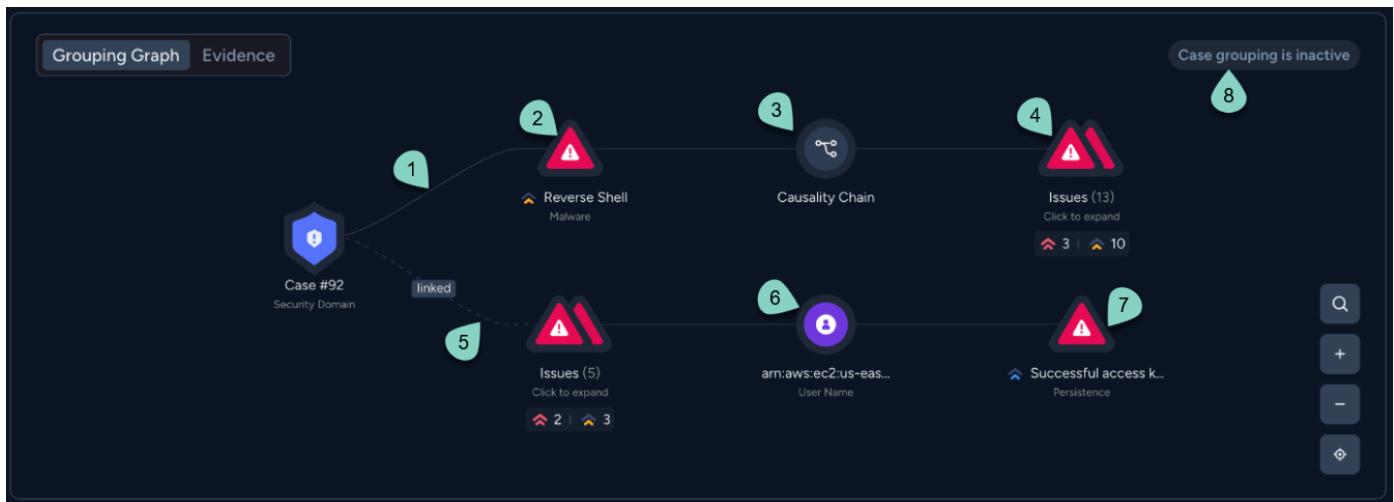
#### Explore the graph

You can interact with the graph to uncover deeper layers of data without leaving the case view:

- **Expand and break down:** Click elements within the graph to expand clusters and view additional node details, such as severity, domains, and current status.
- **Review issues and artifacts:** Hover over any entity in the graph to open a quick-view panel containing high-level details such as severity, domain, and current status. Hover over a cluster to see a breakdown of the severities contained within it.
- **Deep dive into issues:** Click an issue node and select **Open Issue** to view a detailed issue card with granular details about the issue.

Example 61.





The following table breaks down the components in this example:

Label	Explanation
1	Solid edge linking the case node to the issue that initiated case creation.
2	The issue that initiated case creation.
3	Casualty chain related to the initial issue.
4	Cluster of issues. These issues are part of the same causality chain as the initial issue. You can see that there are 13 issues in the cluster, and their severity breakdown.
5	Broken edge linking to a cluster of issues that were manually linked to the case. This is indicated by the linked label.
6	User name related to one or more issues in the linked issues cluster.
7	Issue related to the user name.
8	Case grouping is inactive label. This indicates that the case is no longer accepting new matching issues, which happens when a case grouping threshold is met. For more information, see Case thresholds.

#### 4.3.4.2 | Evidence

##### Abstract

Review the Evidence section of the Case card to see details of causalities and events.

The **Evidence** tab allows you to trace issue causality chains to follow the sequence of events from the initial root cause to the final recorded activity. If causalities are not available, the Events table lists related events for the process node which matches the issue criteria that were not triggered in the issues table, but are informational.

By mapping these dependencies, you can pinpoint exactly how a threat entered your environment and identify the specific actions taken at each stage of the attack. This insight helps you move beyond seeing what happened to understanding the attacker's path, enabling you to implement more effective containment and remediation strategies.



The causality chains are listed according to the Causality Group Owner (CGO), expand the CGO card you want to investigate. Each CGO card displays the CGO name, the following CGO event details, and the causality chain:

- CGO name
- Issue sources associated with the entire causality chain
- Execution time of the causality chain
- Number of issues that include the CGO according to severity.

Expand the causality chain to further investigate in the full Causality view. For more information, see [Causality view](#).

#### 4.3.4.3 | Issue feed

##### Abstract

See a chronological visualization of the case lifecycle in the issue feed.

The issue feed provides a chronological visualization of the case lifecycle, highlighting key case information from initial detection to the most recent activity. Key features include:

- **Issue count:** See the total number of issues linked to the case.
- **High level details:** Review issue details in the timeline, or click an issue to open the full issue card. When an issue is resolved, the issue status and title is dimmed.
- **Contextual insights:** View integrated insights directly within the timeline (when available), providing extra layers of intelligence on why specific events were flagged.
- **Unified progression:** Gain immediate clarity on the speed of an attack, helping you distinguish between rapid automated threats and slow-moving lateral movement.
- **Group issues by attribute:** Sort the issues in the timeline with the **Group By** option that allows you cluster issues and insights by selected criteria, such as category, severity, or detection method.

#### 4.3.4.4 | Associated assets and artifacts

##### Abstract

Review the associated assets and artifacts identified in the case

This section displays the technical entities involved in the case, such as endpoints, hosts, IP addresses, and files. Assets and artifacts are organized by class, such as User, Hash, or IP. Malicious artifacts as identified by WildFire are highlighted red.

Hover over an asset or artifact to see key details about the entity. Click on an asset to see full details in the asset card.

To investigate further, click Ask AI next to an asset or artifact to open the **Agentic Assistant** with an automatically generated prompt tailored to the selected entity. You can also use the **Actions**  menu next to an asset or artifact to drill down to dedicated views or take direct actions on the asset or artifact.

##### NOTE:

If you do not have permissions to access an asset of a case (which is shown as grayed out and locked), check your scoping permissions in [Manage Users](#) or [Manage User Groups](#).

For more information about dedicated asset and artifact views, see [Investigate artifacts and assets](#).

#### 4.3.4.5 | MITRE ATT&CK tactics and techniques

The MITRE ATT&CK card maps observed behaviors to relevant tactics and techniques associated with the issues linked to the case. For increased visibility, click **Insights** to include tactics and techniques from low severity insights.

To see a full breakdown by MITRE ATT&CK tactic and technique, including the number of issues in which a tactic was identified, open the full view.

##### NOTE:

This component is available for cases associated with the Security domain or custom domains.



#### 4.3.4.6 | Detailed View

##### Abstract

Switch to the Detailed View to see a breakdown of case information in a table-based format.

The **Detailed View** in the case card provides a table-based format and custom layouts, ensuring full backward compatibility. You can switch between the **Overview** and the **Detailed View** based on your workflow preferences.

The **Detailed View** supports deep inspection and manual analysis while maintaining access to the same underlying case data. It includes the following tabs:

Tab	Description
Issues & Insights	Displays a list of issues and insights linked to the case. Click on an issue or insight to open the issue card.
Key Assets & Artifacts	Displays asset and artifact information of the key artifacts, hosts, and users associated with the case. Hover over an icon for more information, or click the more options icon to see the available views and actions. For more information about investigating key assets and artifacts, see <a href="#">Investigate artifacts and assets</a> .
Timeline	Displays a chronological representation of issues and actions relating to the case. Each timeline entry represents a type of action that was triggered in the issue.  Issues that include the same artifacts are grouped into one timeline entry and display the common artifact in an interactive link. Click on an entry to view additional details in the Details pane. You can also filter the timeline by action type. Depending on the type of action, you can select the entry to further investigate and take action on it.
Case War Room	The Case War Room is a collection of the Active Response investigation actions, artifacts, and collaboration pieces for an issue or case. It is a chronological journal of the case investigation. You can run commands and playbooks from the War Room and filter the entries for easier viewing.  The War Room facilitates real-time investigation. Powered by ChatOps, the War Room helps you perform different tasks related to their case investigation using CLI commands. For example, running real-time security actions through the CLI, without switching consoles, and running security playbooks, scripts, and commands. For more information, see <a href="#">Use the War Room in an investigation</a>
Executions	Displays the causality chains associated with the case. On this tab, you can investigate a causality chain and take actions on a host. For more information, see <a href="#">Causality view</a> .

##### Investigate issues and insights

The Issues & Insights tab displays a table of the issues and insights associated with the case.

1. Use the toggle to switch between issues and insights, and add filters to the table to refine the displayed entries.
2. Click an issue to open the issue investigation panel. This panel provides detailed information about an issue, enables you to take actions on an issue, open the causality, and start remediation.
3. If required, you can unlink the issue from the case or link it to other related cases. Click the more options icon and select [Manage issue+Link to case](#) or [Unlink from case](#).

##### NOTE:

When an issue is resolved, it remains linked to a case. Once all of the issues in a case are resolved, the case is automatically closed.

##### Run an automation on an issue

You can run or rerun an automation on one or more issues. If there is currently an automation running on one or more of the selected issues, the Run Automation option does not appear. If an automation is running on the issue, but has been paused (for example, waiting for a user action), you can select to rerun the automation or select a new automation.

1. In the Issues & Insights tab, right-click one or more issues and click [Run Automation](#).
2. If the issues have an automation already assigned, choose [Rerun current Automation](#) or [Choose another Automation](#). If the playbooks do not have an automation assigned, select a action to run and define the action parameters.



### 3. Run the automation.

#### Investigate key assets and artifacts

The Key Assets & Artifacts tab displays all the case assets and artifact information of hosts, users, and key artifacts associated with the case.

##### 1. Investigate artifacts.

In the Artifacts section, review the artifacts associated with the case. Each artifact displays, if available, the artifact information and available actions according to the type of artifact: File, IP Address, and Domain.

##### 2. Investigate hosts.

In the Hosts section, review the hosts associated with the case. Each host displays, if available, host information and available actions.

To further investigate the host, select the host name to display the Details panel. The panel is only available for hosts with the agent installed and displays the host name, whether it's connected, along with the Endpoint Details, Agent Details, Network, and Policy information details. If the Details panel is not available, click the more options icon next to a host name to see the available options.

##### 3. Investigate users.

In the Users section, review the users associated with the case. Each user displays, if available, the user information and available actions

#### Investigate the case timeline

The Timeline tab is a chronological representation of issues and actions relating to the case.

##### 1. Navigate to the Timeline tab and filter the actions according to the action type.

##### 2. Investigate a timeline entry.

Each timeline entry is a representation of a type of action that was triggered in the issue. Issues that include the same artifacts are grouped into one timeline entry and display the common artifact in an interactive link. Depending on the type of action, you can select the entry, host names, and artifacts to further investigate the action:

- Locate the action you want to investigate:
  - For Quick Actions and Case Management Actions, you can add and view comments relating to the action.
  - For Issues, click the action to open the Details panel. In the panel, go to the Issues tab to view the issues table filtered by issues ID, the Key Assets to view a list of Hosts and Users associated to the issue, and an option to add Comments.
- Select the Host name to display the endpoint data, if available.
- Select the Artifact to display the following type of information:
  - Hash artifact: Displays the Verdict, File name, and Signature status of the hash value. Select the hash value to view the Wildfire Analysis Report, Add to Block list, Add to Allow list and Search file.
  - Domain artifact: Displays the IP address and VT score of the domain. Select the domain name to Add to EDL.
  - IP address: Display whether the IP address is Internal or External, the Whois findings, and the VT score. Expand Whois to view the findings and Add to EDL.
- In action entries that involved more artifacts, expand Additional artifacts found to further investigate.

#### Investigate case executions

The Executions tab displays all the causality chains associated with the case. The causality chains are aggregated according to the following types of groupings:

- Host Name
  - Host with an agent installed
  - Host without an agent installed
  - Multiple Hosts
  - Undetected Host
- User Name
  - Username
  - Multiple Users
  - Undetected Users



**NOTE:**

- Cloud-related issues are displayed in the User Name grouping.
- Prisma Cloud Compute issues are displayed in the Host Name grouping.

How to investigate case executions

1. Investigate the host causality chains.

In the Executions section, review the hosts associated with the case. Review the host information and click the more options icon to perform actions on the host, or open related views.

2. Investigate a causality chain.

The causality chains are listed according to the Causality Group Owner (CGO), expand the CGO card you want to investigate. Each CGO card displays the CGO name, the following CGO event details, and the causality chain:

- CGO Name
- Issue Sources associated with the entire causality chain
- Execution time of the causality chain
- Number of issues that include the CGO according to severity.

Expand the causality chain to further investigate and perform available Causality View actions. For more information, see [Causality view](#).

#### 4.3.5 | Resolve the case

Abstract

You can start remediating a case by reviewing the actions in the Resolution Center.

After analyzing a case, you can start remediation in the Resolution Center. This process involves executing specific tasks to address the problems identified in the case. Once the remediation tasks are completed and verified, you can officially close the case to reflect its updated status and maintain an accurate audit trail.

##### 4.3.5.1 | Resolution Center

Abstract

Review the remediation action in the Resolution Center to start resolving a case.

The **Resolution Center** is the primary workspace for managing and resolving cases. With a focused, action-oriented flow, you can focus on resolving the entire case rather than investigating isolated issues. By removing fragmented navigation, this workspace allows you to work without context switching, enabling you to open and run playbooks within the case context and quickly review the status of all tasks for all issues in the case.

The **Resolution Center** guides you toward resolution by answering the question, **What should I do next?** You can track your progress using four specialized tabs:

Pending

This tab acts as your to-do list for case actions. It displays any tasks waiting for execution or playbook tasks that require your input.

- **Task details:** You can view the task summary, assignee, and SLA. If SLAs are configured, tasks are sorted by deadline; otherwise, they appear in the order they were created.
- **Play book execution:** Each task shows its source (Issue ID and Automation Name). You can click the Issue ID to open the issue card or click the playbook to open the workplan.

If the playbook is already in progress but requires user input, the label shows the status of the playbook. Click the label to open the Workplan and directly execute the playbook task.

**NOTE:**

Tasks that are already In Progress but still require your input will appear in both the Pending and In Progress tabs.

Recommended

Lists suggested playbooks and response actions to help remediate issues linked to the case.



- **Task details:** You can see details of the recommended tasks, including the name of the source that triggered the recommendation.
- **Consolidated tasks:** If the same action is recommended for multiple issues, it is only listed once. Review the labels on a task to see the issues for which the task is relevant.
- **Playbook execution:** Click a playbook to preview and execute it in the **Work Plan**. If the playbook applies to multiple issues, you can choose which issues to run it against.
- **Recommended response actions:** Click a recommended action to open a dialog with detailed steps for executing the action.

#### In Progress

Track currently running automations and remediation workflows in real time.

- **Real-time tracking:** This tab shows all active playbooks, including those in the run queue.
- **Status details:** Each record includes the playbook name, related issue, and current status (Error, Waiting, or Running).
- **Navigation:** You can click any playbook to open the Work Plan or click an Issue ID to view the associated issue card.

#### Done

This tab provides a clear audit trail of your resolution steps.

You can review a list of completed playbooks and actions. Each record includes the playbook name, the related issue, the completion time, and the final status.

#### 4.3.5.2 | Collaborative notes and comments

Located within the **Resolution Center**, the **Notepad** and **Comments** panels enable team-wide communication and documentation. This workspace ensures all analysts stay aligned by maintaining a continuous record of the investigation.

Capabilities include:

- **Notepad:** Record critical evidence, observations, and investigative steps to maintain a shared history for the case.
- **Comments:** Share progress updates and discuss the case with team members in real-time.

#### 4.3.5.3 | Resolve a case

You can resolve a case in the following ways:

- Manually on the Cases page:
  - Click the case status and select Resolved.
  - In the Resolve case dialog, select the resolution reason and leave a comment.
  - Select whether to resolve all of the issues in the case, and whether to create an exclusion.
  - Click Resolve.
- In the War Room or as a playbook task. Run the `!setParentIncidentFields` command.
- In the API, run the `Update Case` command .

#### **NOTE:**

If a case is resolved with the status `Resolved - Auto Resolved`, Cortex AgentiX can reopen the case for up-to six hours if a new issue is triggered that matches the case. The six-hour period is defined by the timestamp of the last issue that was grouped into the case. After the six-hour period, any new issues are linked to a new case for a new investigation.

#### 4.3.5.4 | Resolution reasons for cases and issues

##### Abstract

Describes the resolution reasons for cases and issues.

When you resolve a case or issue, you must also specify a resolution reason. The following table describes the resolution reasons for selection.

#### **NOTE:**



Resolution Reason	Description
Resolved - True Positive	<p>The case or issue was correctly identified by Cortex AgentiX as a real threat, and the case was successfully handled and resolved.</p> <p><b>NOTE:</b></p> <p>Cases and issues resolved as True Positive and False Positive help Cortex AgentiX to identify real threats in your environment by comparing future cases and associated issues to the resolved cases. Therefore, the handling and scoring of future cases is affected by these resolutions.</p>
Resolved - False Positive	<p>The case or issue is not a real threat.</p> <p><b>NOTE:</b></p> <p>Cases and issues resolved as True Positive and False Positive help Cortex AgentiX to identify real threats in your environment by comparing future cases and associated issues to the resolved cases. Therefore, the handling and scoring of future cases is affected by these resolutions.</p>
Resolved - Security Testing	<p>The case or issue is related to security testing or simulation activity, such as a BAS, pentest, or red team activity.</p>
Resolved - Known Issue	<p>The case or issue is related to an existing issue or an issue that is already being handled.</p>
Resolved - Duplicate Case	<p>The case or issue is a duplicate of another case.</p>
Resolved - Risk Accepted	<p>The case or issue is related to a known mitigation or impact.</p>

If you created a custom resolution, it is also available for selection.

#### 4.3.6 | Use Cortex Agentic Assistant chat in an investigation

##### Abstract

Learn how to use the Agentic Assistant chat.

The Cortex Agentic Assistant chat provides an interactive and intelligent way to simplify and streamline complex security operations. Enter a prompt using natural language, and your agent plans and executes the most relevant actions to fulfill your request.

**NOTE:**

The Cortex Agentic Assistant is currently available in limited regions. For more information see Cortex Agentic Assistant If your tenant is not within one of those regions, you have access to the Cortex Assistant.

To enable the Cortex Agentic Assistant, go to Settings â Configurations â General â Server Settings â Agentic Assistant.

The chat leverages your personal context (such as your name, email, and roles), the agent's description, available actions, and conversation context to enable highly informed and personalized interactions. You can manage multiple chats simultaneously and easily switch between the agents you have access to. Before acting, the agent generates a plan, verifying each step while executing the sequence of actions that fulfill your request.

Accessing and exiting the Agentic Assistant chat is designed to be quick and seamless, allowing you to jump into your investigations or step away with ease.

##### Access the chat

To access the chat, you must have the correct permissions. For more information, see Agentic Assistant role-based access control.

To open the chat window, from the side menu click Agentic Assistant, or from the AgentiX Command Center dashboard click Start Investigation.



## Exit the chat

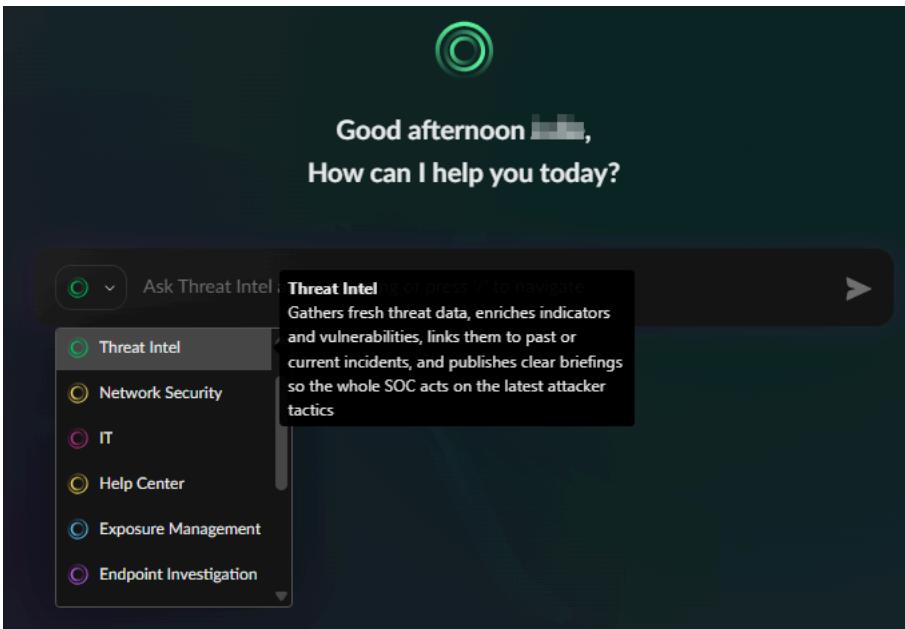
To close the chat window, click anywhere outside the chat window's boundaries or in the side menu click Agentic Assistant.

## Choose an agent

Before you dive into your investigation, select the most relevant AI agent for the job. Each agent is designed with specific goals and functions to help you address different aspects of security operations.

Within the chat prompt, click the agent icon; a list of available agents appears. As you hover over each agent, a brief description pops up explaining what that agent does and its primary focus. Select the agent that best suits your current task or investigation.

You can choose from system agents, public agents other users have created, or agents you have personally built and configured.



## System agents

System agents are pre-built, mission-focused virtual personas provided out-of-the-box by Cortex AgentiX to handle specific security use cases without requiring manual configuration. You can enable or disable system agents, but you cannot edit their core instructions or delete them.

Unlike custom agents which can be private, system agents are available to all users in the tenant, though the specific actions they can execute depend on user permissions. System agents come with defined roles and permissions, for example, the Threat Intel agent is pre-configured to enrich indicators, while the Help Center agent is designed specifically to retrieve documentation.

You can access additional system agents by enabling specific modules or licenses. Ensuring you have the relevant licenses active (for example, Cloud Posture or XSIAM Enterprise) will ensure the corresponding agents appear in your list. For instance, the Exposure Management agent helps prioritize risks but explicitly requires the Exposure Management add-on to function.

If a system agent is missing from your chat, it may be disabled or not included in your license. Go to the Agents Hub, where you can view a list of all enabled and disabled agents (accessible via the side panel in the Agentic Assistant menu). An administrator may need to re-enable it to make it visible in your chat again.

## Examples of system agents

Here are some examples of the specialized system agents, each relevant for specific security workflows:

Agent Type	Description
IT	Automates identity lifecycle enforcement, real-time containment on endpoints and networks, vulnerability and patch governance, asset intelligence upkeep, and end-to-end incident workflow coordination—delivering policy-driven remediation across the enterprise.
Email Investigation	Automates the full lifecycle of email-borne threat response, spanning mailbox search, forensic collection, analysis, containment, and incident closure across all major mail platforms and security layers.
Threat Intel	Gathers fresh threat data, enriches indicators and vulnerabilities, links them to past or current incidents, and publishes clear briefings so the whole SOC acts on the latest attacker tactics.



Agent Type	Description
Help Center	Access Palo Alto Networks' product documentation for additional product support.
Network Security	Audits next-gen firewalls for vulnerabilities, expired certificates, outdated software, risky or unused rules, capacity limits, and other misconfigurations. It searches logs for threats and then automates or guides clean-ups and upgrades to keep the network secure.
Endpoint Investigation	Unifies host-level containment, forensic collection, and remediation across all major EDR/XDR platforms while feeding evidence and status into the SOC's ticketing and collaboration stack.

#### Safeguards for chat security and control

Cortex Agentic Assistant implements the following safeguards to ensure agent plans and executions are secure, approved, and maintains your control over critical system changes.

- Agents are designed to intelligently validate their proposed plans, ensuring that all necessary permissions are in place before any action is taken.
- Cortex Agentic Assistant clarifies ambiguous prompt intentions and blocks requests that may be exploitative or harmful, for example, to perform a malicious operation.
- For any sensitive actions, agents will always require your explicit approval.
- Cortex Agentic Assistant clarifies ambiguous prompt intentions to verify the intention.

#### Engage with your agent

After choosing an agent, in the chat prompt, type a request using natural language. Be as clear and specific as possible. Submit your request by pressing Enter or clicking the submit arrow. Some agents provide relevant chat conversation starters under the chat prompt.

During a conversation, when an agent is formulating a plan or executing steps, clicking the agent will show which actions it is using. You can scroll between the actions or close the panel.

#### TIP:

If you need to quickly access various product pages within Cortex AgentiX, type / in the prompt.

#### Prompt examples and best practices

##### Examples

Using chat prompt conversation starters in the Agentic Assistant simplifies and speeds up your interactions by providing pre-defined, common queries that guide you to relevant actions and information.

For example, a SOC analyst may see the following conversation starters under the chat prompt:

- What are the top issues I should prioritize today?
- Show me all issues with an overdue SLA
- Which automations are waiting for my input?

##### Best practices

To maximize the effectiveness and accuracy of your interactions with Cortex Agentic Assistant, follow these recommended best practices when entering your prompts.



- Be clear and specific

Clearly state your objective and provide the necessary context. Instead of "Fix the issue," try "Investigate issue 1234 and isolate any affected hosts on which malware has been identified." Specify exact values, IDs, and relevant details.

- Break down complex tasks

For multi-step or intricate processes, break your request into smaller steps within a prompt. This allows the agent to focus, validate each step, and helps guide the flow.

- Include key information

Always include relevant incident IDs, indicator values (such as IP addresses and file hashes), or entity names directly in your prompt. The more precise the initial information, the better the agent can leverage its actions and context.

- Specify a desired output or action:

If you need a particular type of output (for example, "Summarize the findings," "List all affected assets") or a specific action (for example, "Isolate host X," "Block IP Y on firewall"), explicitly state it. This helps the agent understand your intent beyond just identifying the problem.

#### Review agent responses

##### **View the plan**

Cortex Agentic Assistant operates with transparency. The agent's proposed plan or steps for any action are always visible.

Click Plan and expand the chevron to review the detailed breakdown of what the agent intends to do.

##### **NOTE:**

An agent's proposed plans and results may contain inaccuracies or errors. Always review the results carefully to ensure you fully understand the proposed action before proceeding.

##### **Navigate long responses**

If an agent's response is long, you can easily scroll through it. To jump directly to the last line of a long response, click the anchor icon.

##### **Start fresh**

Sometimes, an investigation takes a new direction, or you want to pivot to a different task. You can always open a new conversation or start a new investigation path with a new agent whenever needed, giving you maximum flexibility.

##### **Processing time**

While an agent is processing a prompt, you can begin typing a new prompt. However, you can only submit this new prompt once the previous one has completed its processing. For complex actions, the system may indicate that it's taking some time. Be aware that actions exceeding five minutes will result in an error.

##### **Privacy and transparency**

Your conversations within the Agentic Assistant chat are private. However, for transparency and auditing purposes, Cortex AgentIX audit logs record all actions performed by the agents in response to your prompts. This ensures transparency by providing a detailed, traceable record of who initiated an action, what action was taken, and when, without logging the private content of your prompts themselves.

##### **Chat history**

Cortex Agentic Assistant helps you keep track of your investigations by organizing your chat history for easy review and continuity.

Your chat history is listed to the left of the prompt, making it simple to navigate past conversations. The chat history is organized by periods: Chats from today, yesterday, the last seven days, and older. To continue a previous investigation or review a past conversation, scroll through the list and click on the chat you wish to resume.

##### **Manage your chats**

By default, the first prompt you enter in a new chat becomes its title in the history. To edit the chat title or delete a chat that is no longer relevant, click  and select Edit or Delete.

#### **4.3.7 | Create a case**

##### **Abstract**

You can manually create a new case, assign it to a specific domain, and define custom fields for the case.

##### **NOTE:**

To create a case manually, you must have View/Edit permission for Cases and Issues selected under Settings  Configurations  Access Management  Roles  Components  Cases & Issues.



You can create a case directly from the Cases page.

1. On the Cases page click New Case.
2. Under Case Details, specify the name, severity, and (Optional) description.

The severity of a manually generated case cannot be low.

**NOTE:**

You can assign a case to a single domain only, and you cannot change the assigned domain. For more information, see [Case and issue domains](#).

3. (Optional) Under Case Fields, select custom case fields.

Cortex AgentIX validates the Host IP, Local IP, and Remote IP fields.

If you select Set fields as default for new <domain> domain cases, the custom case fields that are configured are saved for all users. When a user next creates a case for the same domain, these fields are automatically configured instead of the default field set.

To reset the custom fields to the system default, click Restore Default Field Set.

4. Under Issue Details, select the issues to link to the case, or create a new issue.

**TIP:**

The issues that you link to a case can be linked to multiple cases, and the issue domains do not need to match the case domain.

5. Under Issue Fields, define the following:

**NOTE:**

This option is only relevant for certain domains.

- MITRE ATT&CK tactics and techniques to assign to the case.
- Custom issue fields.

6. (Optional) Under Playbook, specify playbook run settings. By default, a playbook is run Automatically by trigger.

**NOTE:**

This option is only relevant for certain domains.

7. Click Create new case.

Each case creation generates one issue. The name, the severity, and the description of the generated issue mirrors the name, the severity, and the description of the case.

**NOTE:**

You can't attach files to manually created cases.

## 4.4 | Investigate issues

Abstract

Cortex AgentIX generates issues to bring your attention to security risks in your framework.

Issues help you to monitor and control the security of your system framework by notifying you about risks to security in your framework. Cortex AgentIX generates issues from the following:

- Rules that you set up, such as correlation rules.
- Agents
- Firewalls
- Analytics
- Integrations

Integrations enable you to ingest events, such as phishing emails, SIEM events, from third-party security and management vendors. You might need to configure the integrations to determine how events are classified as events. For example, for email integrations, you might want to classify items based on the subject field, but for SIEM events, you want to classify by event type.

### 4.4.1 | Overview of the Issues page

Abstract

The Issues page consolidates all non-informational issues from your detection sources.



The Issues page consolidates all non-informational issues from your detection sources. By default, the Issues page displays the security issues received over the last seven days. To access the Issues page, go to Cases & Issues → Issues.

Each issue is linked to one or more cases. A case provides the full story of a problem by linking related issues, assets, and artifacts in one place. To make sure that you understand the full picture of how an issue fits into the bigger picture, we recommend that you start your investigation from the Cases page. You can see the issues linked to a case in the Issues & Insights tab of the selected case. Click on an issue to open the Issue card. For more information, see Issue card.

For issues associated with the Health domain, these issues are not linked to cases and should be investigated individually. You can also see Health domain issues on the Health Issues page. For more information, see About health issues.

#### **NOTE:**

Every 12 hours, the system enforces a cleanup policy to remove the oldest issues once the maximum limit is exceeded. The default issue retention period in Cortex AgentIX is 186 days.

Standardized format of user names in issues

Cortex AgentIX processes and displays the names of users in the following standardized format, also termed “normalized user”.

`<company domain>\<username>`

As a result, any issue triggered based on network, authentication, or login events displays the User Name in the standardized format in the Issues and Cases pages. This impacts every issue for Analytics and Cortex AgentIX Analytics BIOC, including Correlation, BIOC, and IOC issues triggered on one of these event types.

Deduplicated FW issues

To reduce noise in your environment, if firewall issues with the same name and host are raised within 24 hours, the issues are deduplicated. A label indicates the number of deduplicated issues up to 1,000 issue counts, larger quantities display as 1000+.

Issue fields

To see a full list of issue fields and descriptions, run the following query in the Query Builder:

```
datamodel dataset = issues
```

#### 4.4.2 | Issue card

Abstract

On the Issue card, you can see details of the selected issue and take actions on an issue.

The Issue card provides a full breakdown of an issue, helping you understand the root cause and take action through relevant evidence, remediation guidance, and response options.

The issue card supports full case investigation by retaining case context. Once you have finished reviewing an issue, close the card to return to the initial case investigation.

Each issue card adapts to the type of issue you’re investigating, surfacing the most relevant information and tools at every stage of the workflow. While layouts may vary, most issues share a common set of tabs designed to support triage, investigation, and resolution.

Tab	Description
Overview	<p>Displays a description of the issue and provides key information, including:</p> <ul style="list-style-type: none"><li>• Assignee</li><li>• Status</li><li>• Time at which the issue was created and updated</li><li>• Suggested automations to run on the issue. Click the automation to open to the Work Plan tab with details of the automation.</li><li>• Affected Assets with links to the affected asset cards</li><li>• Cases linked to the issue</li></ul> <p>The Evidence section contains information to help you investigate the issue, such as the causality chain.</p> <p><b>NOTE:</b></p> <p>This section is context-specific and shows data according to the issue context.</p>



Tab	Description
Issue Information	<p>Displays a summary of the issue, such as issue details, indicators, and outstanding tasks. Some fields are informational and some can be edited. Includes the following sections (depending on the layout):</p> <ul style="list-style-type: none"> <li>• ISSUE DETAILS: A summary of the issue, such as type, severity, and when the issue occurred. You can update these fields as required.</li> <li>• COMMAND AND TASK RESULTS: Lists any manual commands and playbook task results.</li> <li>• WORK PLAN: View or take action on the following: <ul style="list-style-type: none"> <li>◦ Playbook tasks: When a playbook runs, any outstanding tasks appear. You can take various actions here or in the Work Plan tab.</li> <li>◦ To-Do Tasks: An ad-hoc item that is not attached to the Work Plan. Create tasks for users to complete as part of an investigation. These are like a To-Do list that you keep in an investigation on an ad-hoc basis, rather than the Work Plan, which follows a pre-defined process. You can view or create To-Do tasks.</li> </ul> </li> <li>• NOTES: Helps you understand specific actions taken, and allows you to view conversations between analysts to see how they arrived at a certain decision. You can see the thought process behind identifying key evidence and identifying similar cases.</li> <li>• MALICIOUS OR SUSPICIOUS INDICATORS: A list of any malicious or suspicious indicators. If you have the Threat Intel add-on, you can pivot to the Indicators page, where you can take further action on the indicator.</li> </ul> <p><b>LICENSE TYPE:</b></p> <p>Requires the TIM add-on.</p> <ul style="list-style-type: none"> <li>• INDICATORS HANDLING: Take actions on indicators from the displayed options.</li> </ul>
Technical Information	Displays an overview of the information collected about the investigation, such as indicators, email information, URL screenshots, etc. When you run a playbook, the sections are automatically completed.
Investigation Tools	Enables you to take action on the issue, such as converting a JSON file to CSV and checking if the IP address is in CIDR.
War Room	A comprehensive collection of all investigation actions, artifacts, and collaboration. It is a chronological journal of the issue investigation. Each issue has a unique War Room. For information, see Use the War Room in an investigation.
Work Plan	A visual representation of the running playbook that is assigned to the issue. For more information, see Use the Work Plan in an investigation.
Actions	Recommended actions to resolve the issue.

#### 4.4.3 | Link or unlink issues from a case

You can link and unlink issues from cases. An issue can be assigned to more than one case, and the case domain can be different from the issue domain.

##### Link issues to a case

From the Issues page, select one or more issues that you want to link, right-click and select Manage Issue+Link to case. You can select one or more case to link the issues.

##### Unlink an issue from a case

From the Issues page, select the issue that you want to unlink, right-click and select Manage Issue+Unlink from case. You can select one or more cases to unlink the issue. You cannot bulk select issues to unlink.

#### 4.4.4 | Run an automation on an issue

##### Abstract



Save time and expense by using playbooks and Quick Actions to automatically investigate and take remedial action on issues.

You can automate issue investigation and remediation by running a playbook or Quick Action on one or more issues. Automations can help to improve efficiency by automating and standardizing your workflows, promoting consistent and effective case response and management. For example, automations can automatically remediate a case by interacting with a third-party integration or open tickets in a ticketing system such as Jira.

You can view the playbook that is running on an issue or the playbooks that have already run in the Work Plan for an issue. You can view Quick Actions in the War Room for an issue.

**NOTE:**

In addition to automation, some playbooks contain manual tasks that prompt the analyst for input. This enables you to enhance an automation workflow with analyst input.

You can run automations in the following ways:

Manually run a playbook or Quick Action on one or more issues

1. Right-click one or more issues in the Issues table and select Run Automation.

If there is currently an automation running on one or more of the selected issues, the Run Automation option does not appear. If an automation is running on the issue, but has been paused (for example, waiting for a user action), you can select to rerun the automation or select a new automation.

2. If the issues have an automation already assigned, choose Rerun current Automation or Select another Automation. If the issues do not have an automation assigned, Select Automation.
3. If you are not rerunning the current assigned automation, select an automation to run for the selected issue(s).
4. Click Run.

**NOTE:**

You can also manually select a playbook to run from the Issue Work Plan tab.

Apply automation rules

You can create automation rules that automatically run a playbook or Quick Action when an issue is created that meets specific criteria. For more information, see Create an automation rule.

For more information, see Automation in Cortex AgentiX.

#### 4.4.5 | Use the War Room in an investigation

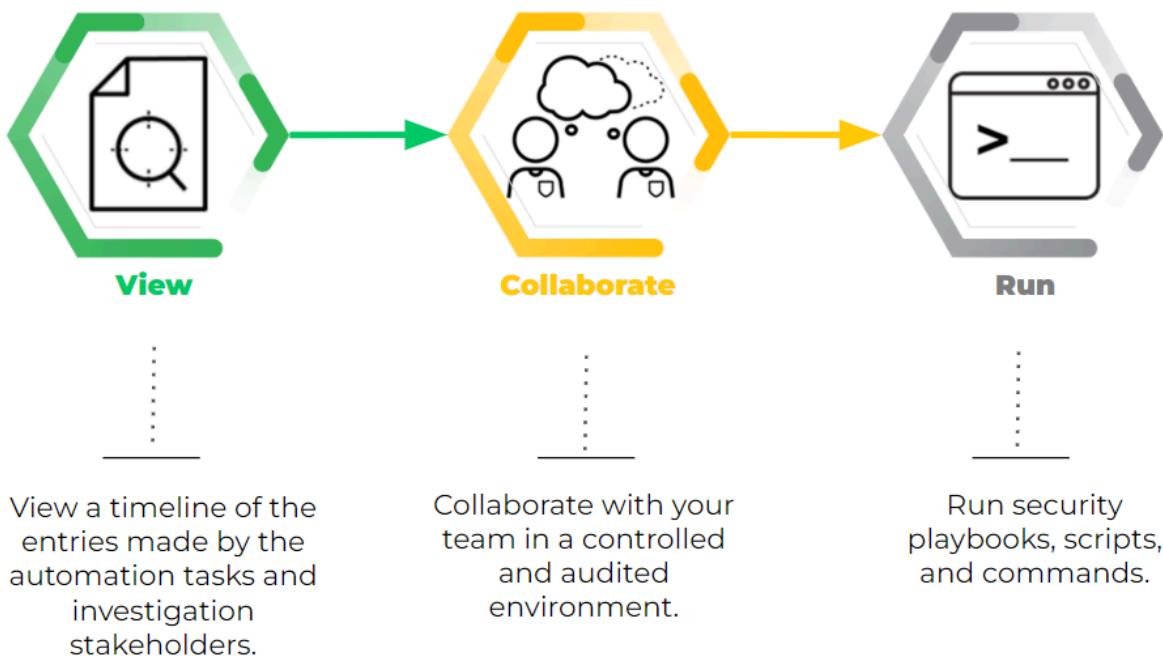
Abstract

Use the War Room for real-time investigation into a case, to filter war room entries, and to disable indicator notifications.

The War Room contains an audit trail of all automatic or manual actions that take place in a case or issue. A War Room is where you can review and interact with your case or issue. Cortex AgentiX provides machine learning insights to suggest the most effective analysts and command-sets. Each case and issue has a unique War Room.



# The War Room: A Chronological Journal



Within Cortex AgentiX, real-time investigation is facilitated through the War Room, which is powered by ChatOps. In the War Room you can take the following actions:

- Run real-time security actions through the CLI, without switching consoles
- Run security playbooks, scripts, and commands
- Collaborate and execute remote actions across integrated products
- Capture case context from different sources.
- Document all actions in one source.
- Communicate with others for joint investigations.

## NOTE:

The case War Room is usually used for communication capabilities, but unlike the issue War Room, it does not include playbook specific entries. The case War Room enables you to investigate an entire case, not just an issue.

Every case has a War Room, but every user has access, subject to permissions, to a private War Room called the Playground.

## The Playground

The Playground is a non-production environment where you can safely develop and test data, such as scripts, APIs, and commands. It is an investigation area that is not connected to a live (active) investigation.

To access the Playground, do one of the following:

- Go to Investigation & Response → Automation → Playground
- In any browser, type <https://<tenant>.<region>.paloaltonetworks.com/playground>

## TIP:

In the Playground, you can clear the context data, if needed, which deletes everything in the Playground context data, but does not affect the actual issue or case. To clear the context, run `!DeleteContext all=yes` from the CLI or click Clear Context Data while viewing the context data.

## The War Room

When you open the War Room, you can see all the actions taken on a case, such as commands and notes in several formats such as Markdown, and HTML. When Markdown, HTML, or geographical information is received, the content is displayed in the relevant format.

To view specific data entries, you can filter entries by selecting the relevant checkbox, such as:



- Chats: Shows communication between team members.
- Notes: Any entries marked as notes.
- Files: Anything uploaded to the War Room in a playbook, script, or by the analyst.
- Issue History: Any issue field that was modified.
- Commands and playbook tasks: Any actions taken by playbook tasks or run manually by the analyst.
- Tags: Any tags added to the investigation.

**NOTE:**

Cortex AgentIX does not index notes and chats.

In each War Room entry, you can take the following actions:

Action	Description
Mark as note	<p>Marks the entry as a note, which can help you understand why certain action was taken and assist future decisions.</p> <p>You can also add a note by doing the following:</p> <ul style="list-style-type: none"> <li>• Upload a file to the War Room by selecting Mark as Note.</li> <li>• If the Issue Overview tab includes a NOTES section, add it to the section.</li> <li>• In a playbook task (Advanced tab)</li> </ul> <p>Tasks can be automatically added from script outputs as notes.</p> <ul style="list-style-type: none"> <li>• In the CLI by running the <code>!markAsNote entryIDs=&lt;ID of the war room entry&gt;</code> command.</li> </ul> <p>In the relevant War Room entry, click Copy to CLI to retrieve the <code>ID of the War Room entry</code>.</p> <p>When marked as a note, it is highlighted, so you can easily find them in the War Room or the Issue Overview tab.</p>
View artifact in new tab	Opens a new tab for the artifact.
Detach from task	Removes a task from the artifact.
Attach to a task	Adds a task to the artifact.
Add tags	Add any relevant tags to use that help you find relevant information.
Copy to CLI	<ul style="list-style-type: none"> <li>• ID: Entry IDs are used to uniquely identify War Room entries and take the format <code>&lt;ENTRY_IDENTIFIER&gt;@&lt;CASE_ID&gt;</code>, for example, <code>54925dc3-a972-4489-8bef-793331fa6c77@1</code>. Many out-of-the-box commands and scripts use entry IDs arguments to pass in files as inputs.</li> <li>• URL: Copy the URL which is a direct link to the War Room entry</li> </ul> <p>To find the entry ID or URL of an entry in the War Room, click on the vertical ellipsis icon at the upper right of the entry, then copy the value.</p>

**Run Commands in the War Room CLI**

Cortex AgentIX enables you to run system commands, integration commands, and scripts from an integrated command line interface (CLI), which enables you to make comments in your case (in plain text or Markdown) and to execute automation scripts, system commands, and integration commands. This gives SOC teams the power to execute automations ad-hoc to support their investigations or make notes as they investigate cases.

In the CLI, you can run various commands by typing the following:



Action	Description
!	Runs integration commands, scripts, and built-in commands, such as adding evidence and assigning an analyst.

You can find relevant commands, scripts, and arguments with the CLI's auto-complete feature. This also includes fuzzy searching to help you find relevant commands based on keywords. If you type the exclamation mark (!) and start typing, autocomplete populates with options that might suit your needs. For example, if you want to work with tasks, type !task, and all commands and scripts that include the task in their name will display.

#### TIP:

You can use the up/down arrow buttons in the CLI to do a reverse history search for previous commands with the same prefix.

Special characters

Characters	Description
&&,   , !, {, }, [, ], (, ), ~, *, ?	To use these characters, place them within single or double quotes. An escape character \ is not required.
\, \n, \t, \r, ", ^, :, comma, and space	To use these characters, place them within single or double quotes and use an escape character \.

Common arguments

The following common arguments are available for every script run from the CLI.

Argument Name	Description
auto-extract	Whether/when to extract indicators. Possible values: <ul style="list-style-type: none"> <li><b>inline</b>: Extracts indicators within the indicator extraction run context (synchronously).</li> <li><b>outofBand</b>: Extracts indicators in parallel (asynchronously) to other actions.</li> <li><b>none</b>: Does not extract indicators (recommended for scripts with large outputs when indicator extraction is not required).</li> </ul>
execution-password	Supplies a password to run a password-protected script.
execution-timeout	Defines how long a command waits in seconds before it times out.
extend-context	Select which information from the raw JSON you want to add to the context data.  For a single value: <code>contextKey=RawJsonOutputPath</code>  For multiple values: <code>contextKey1=RawJsonOutputPath1::contextKey2=RawJsonOutputPath2</code>
ignore-outputs	Possible values: <code>true</code> or <code>false</code> . If set to <code>true</code> , it does not store outputs in the context (besides extend context).
raw-response	Possible values: <code>true</code> or <code>false</code> . If set to <code>true</code> , it returns the raw JSON result from the script.
retry-count	Determines how many times the script attempts to run before generating an error.
retry-interval	Determines the wait time (in seconds) between each script execution.



Argument Name	Description
using	Selects which integration instance runs the command.
using-brand	Selects which integration runs the command. If the selected integration has multiple instances, the script may run multiple times. Use the <code>using</code> argument to select a single integration instance.
using-category	Selects which category of integrations runs the command. If the selected category includes multiple integration instances, the script may run multiple times. Use the <code>using</code> argument to select a single integration instance.

Run commands in the Automations browser

You can view and run commands and scripts (not system commands, operations, and notifications) in the Automations Browser, by clicking  next to the CLI.

The Automations Browser enables you to run commands and all associated arguments. The scripts and commands are separated into sections such as scripts and built-in commands. In each argument, you can do the following:

- Hardcode the value
- Use a dynamic value

You can dynamically pass information into the argument by clicking the curly bracket. For example, the `EmailAskUser` command asks a user a question via email. In the `email` argument, rather than typing the user's email address, you can send it to whoever created the case.

1. In the email field, click the curly brackets.

2. In the search box, enter `created`.

3. Under CASE DETAILS click Created by.

The email argument appears as  `${alert.dbotCreatedBy}`.

4. Run the command.

An email is sent to the user who created the case.

You can use transformers and filters to filter and transform data from the command.

Common arguments when using the Automations browser

Argument	Description
Using	Selects which integration instance runs the command.
Extend context	Determines the wait time (in seconds) between each script execution.  For a single value: <code>contextKey=RawJsonOutputPath</code>  For multiple values: <code>contextKey1=RawJsonOutputPath1::contextKey2=RawJsonOutputPath2</code>
Ignore outputs	Does not store outputs in the context (besides extend context).
Execution timeout (seconds)	Defines how long a command waits in seconds before it times out.
Number of retries	Determines how many times the script attempts to run before generating an error.
Retry interval (seconds)	Determines the wait time (in seconds) between each script execution.

Examples using the CLI



To run the print script with a value of "hello" and the key a from the context:

```
!Print value="hello ${a}"
```

To run the Python command returning Hello World using escape characters:

```
!py script="demisto.results(\"hello world\")"
```

To run the Python command returning Hello World using backticks:

```
!py script=`demisto.results("hello world")`
```

#### 4.4.6 | Use the Work Plan in an investigation

##### Abstract

A Work Plan is a visual representation of the running playbook that is assigned to a case. Use it to monitor and manage a playbook workflow.

The Work Plan is a visual representation of the running playbook assigned to the issue. Playbooks enable you to automate many security processes, such as managing your investigations and handling tickets. Work Plans enable you to monitor and manage a playbook workflow, and add new tasks to tailor the playbook to a specific investigation.

In an investigation, when you open the Work Plan tab you can see the playbook, the playbook name, and navigation tools.

By default, the Follow checkbox is checked, which allows you to see the playbook executing in real-time. The playbook moves when a task is completed.

In the Work Plan you can do the following:

Action	Description
Change the default playbook	On the left-hand side of the window, select the playbook you want to run. When changing the playbook, all completed tasks are removed and the new playbook will run. If you select playbooks several times you can view the history of which playbooks ran.
Rerun the playbook	When changing the playbook, select the current playbook to run again.
View inputs and outputs	View the inputs and outputs of each task that has run. You can't view inputs and outputs of any task that hasn't run.
Manage tasks	View, create, and edit a playbook task. For each task, you can do the following: <ul style="list-style-type: none"><li>• Designate tasks as complete either manually or by running a script.</li><li>• Assign an owner.</li><li>• Set a due date.</li><li>• Add comments and completed notes, as required.</li><li>• View any automation exclusion policies that affected the task execution. Automation exclusion policies prevent automated remediation on critical assets specified by admins. The Policies tab only appears if the task includes a command or script affected by an automation exclusion policy.</li></ul> You can manage these tasks in the CLI by using the /task command.
Export to a PNG	Export the Work plan to a PNG format for easy analysis.

##### Example 62.

For a phishing investigation, after the initial playbook run parses the email and extracts email addresses, as part of the manual investigation, you could use the Email Address Enrichment - Generic v2.1 playbook as an ad-hoc playbook task to get more information about these email addresses.

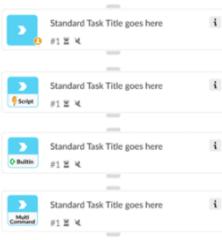


The color coding and symbols in the Work Plan help you to easily troubleshoot errors or respond to manual steps. The following table displays the playbook tasks and icons in the Work Plan.

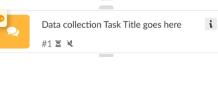
**IMPORTANT:**

A playbook will not continue its execution path if a prior task has failed; you must resolve the failed task before subsequent tasks can run.

Playbook tasks and icons in the Work Plan

Task	Description
	<p> <b>Standard manual task</b></p> <p>An arrow with a light blue square background indicates a standard manual task. The following are kinds of standard tasks.</p> <ul style="list-style-type: none"> <li>• Manual Standard task (no lightning bolt logo):</li> </ul> <p>These tasks are used where usually it's not possible to automate them. You can add comments, assign them to an owner, and set a due date. The analyst who is responsible for the investigation needs to complete the task before the Work Plan can continue. A user icon (  ) indicates the task requires manual inputs.</p> <ul style="list-style-type: none"> <li>• Automated Standard task (with lightning bolt script logo):</li> </ul> <p>A single command or script that is set to automatically run when the Work Plan execution reaches this step. Some scripts need arguments in order to run - make sure to set them up properly. If left empty, the analyst who is responsible for the investigation will need to complete them so the script will run and the Work Plan can continue.</p> <ul style="list-style-type: none"> <li>• Automated Standard task (with Builtin logo):</li> </ul> <p>A single system command or script that is set to automatically run when the Work Plan reaches this step. Some scripts need arguments in order to run - make sure to set them up properly. If left empty, the analyst who is responsible for the investigation will need to complete them so the script will run and the Work Plan can continue.</p> <ul style="list-style-type: none"> <li>• Automated Standard task (with Multi Command logo):</li> </ul> <p>A generic single command or script that can be used with multiple integrations is set to automatically run when the Work Plan reaches this step. Some scripts need arguments in order to run - make sure to set them up properly. If left empty, the analyst who is responsible for the investigation will need to complete them so the script will run and the Work Plan can continue.</p>
	<p> <b>Conditional task</b></p> <p>A diamond icon in a purple square background indicates a conditional task used as decision trees in your Work Plan. The following are kinds of conditional tasks.</p> <ul style="list-style-type: none"> <li>• Manual conditional task. A user icon (  ) indicates the task requires manual inputs.</li> <li>• Automated conditional task (with the lightning bolt script logo).</li> <li>• Automated conditional task that uses a system script (with the Builtin logo).</li> </ul>
	<p> <b>Data collection task / Communication task</b></p> <p>The speech bubble in a turquoise background indicates a data collection task. This task prompts the receivers to respond to a multi-question form and submit replies, even if they are not Cortex users. A user icon (  ) indicates the task requires manual inputs.</p>
	<p> <b>Sub-playbook task</b></p> <p>The workflow icon in a blue background indicates that the task is a playbook nested within the parent playbook. You can view the playbook by opening the task and selecting Open sub-playbook.</p>



Task	Description
	<p><b>Task containing an error</b></p> <p>Scripts or sub-playbooks that have errors are designated by a red triangle. You need to open the script or sub-playbook to review the errors.</p>
	<p><b>Task containing a deprecated script or needs to be updated</b></p> <p>Scripts or sub-playbooks that have updates or are deprecated are designated by a yellow triangle. You need to update the scripts, integration commands, or sub-playbook tasks to their most current version.</p>
	<p> <b>Set to skip</b></p> <p>When a task is set to skip, the skip icon will be orange.</p>
	<p> <b>Breakpoint</b></p> <p>When the Work Plan reaches a breakpoint, the task has an orange line at the top to indicate the breakpoint.</p>
	<p> <b>Input / Output Overridden inputs or outputs</b></p> <p>When a task is set to have overridden inputs or outputs, the word Input or Output appears in orange.</p>
	<p> <b>Pending/in queue task</b></p> <p>When the Work Plan starts to run, all tasks that are about to be performed are gray.</p>
	<p> <b>Running/ in progress task</b></p> <p>A spinning circle inside the gray square indicates a running/in progress task.</p>
	<p> <b>Completed task</b></p> <p>The green square indicates a completed task.</p>
	<p> <b>Waiting task</b></p> <p>The orange square indicates that the task is pending action.</p> <p>If you hover over the icon on the top left corner, details about the reason the task is in waiting mode appear.</p> <p>The user icon () indicates the task requires you to open it and manually mark it as complete.</p> <p>A speech bubble icon () indicates the task is waiting for a questionnaire to be completed.</p>
	<p><b>Failed task</b></p> <p>The red warning icon indicates that the task failed to complete as expected and requires manual inspection and troubleshooting. Contact your Cortex AgentX administrator.</p> <p>If you hover on the icon on the top left corner, details about the specific problem appear.</p> <p>If a red warning icon is paired with the clock icon (,), the task's SLA is overdue.</p>



Task	Description
 <p>Standard Task Title goes here #1 / 5</p>	<p><b>Skipped task</b></p> <p>The task will look faded to indicate it was not executed. This can happen if this task was set to be skipped when an error occurs, or if it is in a branch that was not executed if a condition wasn't met.</p>

#### Add ad-hoc tasks to the Work Plan

As part of your issue investigation, within the Work Plan you can create tasks for a specific iteration of a playbook. The task type can be an automation or another playbook. For example, within a manual task, you might need to enrich some data and run an investigation playbook.

When you create a task, add a name, automation, and description. The name and description should be meaningful so that the task corresponds to the data that you are collecting.

1. In the Cases page, select the case to update.
2. In the Issues & Insights tab, click the issue to add the task to and then click the Work Plan tab.
3. In the Work Plan, go to the task where you want to add a new task and click the + sign at the bottom right-hand corner of the task.

The ad-hoc task is added after the task you clicked.

4. Select the task type.
  - Standard: Runs a single automation.
  - Playbook: Runs a playbook to enhance the investigation.

The playbook functions as any playbook would and requires you to define the inputs and outputs, as well as any other details.
5. To run the Work Plan again click the Run Again icon.

#### 4.4.7 | Manage synced tickets in issues

##### Abstract

Manually sync a ticket to an issue

You can set up integrations in Cortex AgentIX that mirror Cortex issues with external applications, such as Atlassian Jira or ServiceNow. When mirroring issues (also referred to as issue syncing), you can make changes in an external application that will be reflected in Cortex AgentIX, and vice versa. If an issue is mirrored with an external application, you have the following options:

- **Link the ticket to the issue:** If an issue is linked to a ticket, the ticket number is displayed in the Overview section of the issue card. You see details about the status of the ticket by clicking on the ticket number.
- **Sync changes between the issue and the ticket:** If an issue is synced to a ticket, changes are synchronized in an outbound, inbound, or bi-directional flow.

##### NOTE:

Multiple tickets can be linked to an issue with outbound syncing. Issues with inbound syncing can be linked to a single ticket only.

##### Manually create a synced ticket

##### PREREQUISITE:

An integration must be configured before you can sync issues. For more information, see Set up an integration for mirroring issues.

You can manually sync existing issues with external applications.

1. From the **Issues** page, right-click an issue and select Run Automation → Select Automation.
2. Under Quick Actions, select the action you want to configure, such as Create Jira Ticket or Create ServiceNow Ticket.
3. Define the required ticket parameters.

##### NOTE:

Using issue fields as variables is not currently supported.



4. Under Using, select the name of the instance to execute the command.

**WARNING:**

If you leave this field blank, all configured instances will be used.

5. Under Sync Configuration, the following options are displayed, depending on your selection:

- Link to issue: select this option if you want the issue to be linked to the created ticket. You must check this option if you want to sync the issue with the ticket.
- Sync Direction: select the syncing configuration:
  - Inbound: Sync changes from the external ticket with the Cortex AgentiX issue.
  - Outbound: Sync changes from the Cortex AgentiX issue with the external ticket.
  - Bi-directional: Sync changes in both directions.
  - None: Do not sync changes between the Cortex AgentiX issue with the external ticket. If you select this option, the tickets are still linked, but changes are not synced. You can update this option at any time to start syncing.
- Select the sync profile you want to use. Cortex AgentiX provides default outbound and inbound sync profiles, or add a custom sync profiles.

For more information about sync profiles, see [Create a sync profile](#).

**NOTE:**

You can only define a single inbound profile. If you change the inbound sync profile the current profile is overwritten.

You can define multiple outbound profiles; one issue can update multiple tickets.

6. Click OK.

After ticket creation, the ticket number is shown in the Issue card. Click on the ticket number to see details about the created ticket and the syncing configuration.

In addition, the execution is recorded in the **War Room** tab. If there is an error in the requested action, you can see details in the audit.

Example 63.

The following example shows an automation run on an issue to create a ServiceNow ticket that is synced in an outbound flow with the ticket.



The screenshot shows the War Room interface. On the left, there's a search bar and a section for 'Quick Actions (15)' containing various options like 'Create Jira Ticket', 'Isolate Endpoint', and 'Run Malware Scan'. Below that is a section for 'Org Playbooks (4)' with three entries: 'jira-allfield-inbound', 'Jira ticket creation', and 'playbooi'. On the right, a large dialog box titled 'SET ACTION PARAMETERS' is open for 'Create ServiceNow Ticket'. It includes fields for 'Description\*', 'Severity\*', 'Short Description\*', 'Ticket Type\*', 'Sync Configuration' (with 'Link to issue' checked), 'Sync Direction' (set to 'Outbound'), and an 'Outbound Profile' dropdown. At the bottom right of the dialog are 'Cancel' and 'Ok' buttons.

Run a War Room command to create and sync a ticket

You can run the following command in the War Room to create an external ticket and define the syncing configuration:

```
!jira-create-issue-quick-action summary=<summary> project_key=<key> issue_type_name=<type>
description=<description> using=<instance> mirroring_link_to_object="true"
mirroring_sync_direction=<syncDirection> mirroring_outbound_profile_id=<profileID>"
```

**TIP:**

You can find a sync profile ID under Settings → Configurations → Object Setup → Issues → Sync Profiles. By default, the ID field is not displayed in the table. Click the three-dot menu and add it to the layout.

Example 64.

The following example creates a Jira Bug ticket for the Project Key SCRUM, with an Outbound sync configuration:

```
!jira-create-issue-quick-action summary="Restrict ingress on AWS Network ACLs for admin ports 22 and 3349"
project_key="SCRUM" issue_type_name="Bug" description="We identified that multiple AWS Network ACLs are
allowing inbound (ingress) traffic on admin ports" using="JiraV3" mirroring_link_to_object="true"
mirroring_sync_direction="OUTBOUND" mirroring_outbound_profile_id="h8e14996-8695-5396-9g87-f08suum907486"
```

Edit or disable ticket syncing

You can change the syncing configuration between a ticket and an issue from the issue card.

1. In the Overview section of the issue card, click on the external ticket number.

A panel opens with details of the ticket.

2. Click on the settings icon.

3. Under Sync Configuration, change the syncing configuration as required.

**NOTE:**

If you change the selected inbound sync profile, the original sync profile is immediately overwritten.



4. To disable ticket syncing, take one of the following actions:

- To pause ticket syncing, set the Sync Direction value to None.

This temporarily stops the tickets from syncing, but the tickets are still linked. You can update the syncing configuration at any time to resume ticket syncing.

- To unlink the tickets, uncheck Link to issue.

This action is not reversible.

5. Click Save.

#### Limitations of issue mirroring

Consider the following limitations of issue mirroring:

- Issue syncing requires the latest version of Atlassian Jira (V3) and ServiceNow (V2).
- Issue syncing is currently supported in Atlassian Jira (V3) and ServiceNow (V2) only.
- You can sync up to 50K objects.
- You can create a maximum of 200 sync profiles.
- Cortex AgentiX supports up-to 100 Inbound syncs across all synced tickets over a two-minute time period. Any additional changes beyond this limit will not be synced.
- If a connector instance is deleted or disabled, tickets are no longer synced and external ticket information is not available.
- Custom statuses are not supported.
- Currently, a specific set of fields is supported.

### 4.4.8 | Issue investigation actions

#### 4.4.8.1 | Copy issues

##### Abstract

You can copy an issue into memory.

You can copy issue text into memory and paste it into an email. This is helpful if you need to share or discuss a specific issue with someone. If you copy a field value, you can also paste it into a search or begin a query.

##### How to copy an issue value

1. From the Issues page, right-click the issue you want to send.

2. Select one of the following options:

- Copy text to clipboard
- Copy entire row
- Copy issue URL

Cortex AgentiX saves the copied text to memory.

3. Paste the URL into an email or use it as needed to share the information.

#### 4.4.8.2 | Investigate contributing events

##### Abstract

You can investigate the events created by an issue.

When investigating an issue generated by a correlation rule, you can view all of the events created for the issue. You can have up to 1000 events per correlation rule.

In addition, if the correlation rule includes a drilldown query you can run the query in the Query Builder. The drilldown query provides additional information about an issue for further investigation.

##### How to investigate contributing events



- From the Issues table, locate an issue created by a correlation rule.
- Right-click the row, and select Manage Issue → Investigate Contributing Events.
- (Optional) Open the drilldown query, if available.  
Right-click the row and select Manage Issue → Open Drilldown Query.

The drilldown query can accept parameters from the issue output for the correlation rule. In addition, the issue time frame used to run the drilldown query provides more details about the issue generated by the correlation rule. The time frame is the minimum and maximum timestamps of the events for the issue. If there is only one event, the event timestamp is the time frame used for the query.

#### 4.4.8.3 | Export issue details to a file

##### Abstract

You can review issue details offline by exporting issues to a TSV file.

To archive, continue investigation offline, or parse issue details, you can export issues to a tab-separated values (TSV) file:

- From the Issues page, adjust the filters to identify the issues you want to export.
- When you are satisfied with the results, click the download icon ().

The icon is grayed out when there are no results.

Cortex AgentiX exports the filtered result set to the TSV file.

#### 4.4.8.4 | Exclude an issue

##### Abstract

You can exclude issues that are not deemed to be a threat.

During the process of triaging and investigating issues, you might determine that an issue does not indicate threat. You can choose to exclude the issue, which hides the issue, excludes it from cases, and excludes it from search query results.

You can also set up issue exclusion rules that automatically exclude issues that match certain criteria. For more information, see Issue exclusions.

##### How to exclude an issue

- From the Issues page, locate the issue you want to exclude.
- Right-click the row, and select Manage Issue → Exclude Issue.

A notification displays indicating the exclusion is in progress.

#### 4.4.8.5 | Query case and issue data

##### Abstract

You can run queries on case and issue data with the **cases** and **issues** datasets.

Cortex AgentiX uses Cortex Query Language (XQL) as the primary language for searching, analyzing, and transforming security data. XQL allows for highly efficient querying across vast amounts of security telemetry, such as:

- Threat hunting: Proactively search your entire environment for malicious activity, anomalies, and indicators of compromise (IOCs). Formulate queries to look for specific patterns of behavior that might indicate an ongoing attack, even if no alert has been triggered.
- Investigation: When a case or issue is generated, XQL allows security analysts to drill down into the underlying data, understand the full scope of an attack, identify affected assets, and trace the attacker's actions.
- Forensics: Extract detailed information about past events for post-incident analysis and compliance audits.
- Reports and dashboards: Create custom reports and dashboards to visualize security posture, track key metrics, and communicate insights to stakeholders.

To view and use sample investigative queries, such as the Top Unresolved High Severity Cases query, go to Investigation & Response → Search → Query Builder → XQL → Query Library. For more information about using XQL, see Cortex AgentiX XQL.

You can query case and issue data in the **cases** and **issues** datasets. When using the **issues** dataset, keep in mind the following:



- Informational issues are not included in this dataset.
- Issue fields are limited to certain fields available in the API. For the full list, see Get Alerts Multi-Events v2 API.

The `issues` dataset is categorized by domain. To query only security issues, use the following XQL:

```
dataset = issues | filter issue_domain = "SECURITY"
```

To query only posture issues, use the following XQL:

```
dataset = issues | filter issue_domain = "POSTURE"
```

#### 4.4.8.6 | Run indicator extraction in the CLI

Abstract

Use reputation commands, `extractIndicators` command, or the `enrichIndicators` command in the CLI.

In the issue War Room CLI you can run the following commands at the issue level to extract and enrich indicators:

- `!extractIndicators`

If you want to extract indicators from non-War-Room-entry sources (such as extracting from files), use the `!extractIndicators` command from the issue War Room CLI. The command does not create indicators but extracts them only. Use the command to do the following:

- Validate regex: Test a specific string to see if the relevant indicators are extracted correctly, such as a URL.
- In a playbook or automation: The command extracts indicators in a playbook or automation non-war-room-source, and potentially also creates and enriches them (if required).

You can extract from the following:

- A specified entry (an entry ID)
- Investigation (Investigation ID)
- Text
- File path

For example, type `!extractIndicators text="some text 1.1.1.1 something" Auto extract=inline`. The entry text contains the text of the indicators, which is extracted and enriched. You can also extract indicators by adding the auto-extract parameter with the script and the mode for which you are setting it up. For example, `!ReadFile entryId=826@101 auto-extract=inline`. Usually, when using the CLI, you want to disable indicator extraction. For example, if you return internal/private data to the War Room, and you do not want it to be extracted and enriched in third-party services, add `auto-extract=none` to your CLI command.

- `!enrichIndicators`

The `!enrichIndicators` command is usually used when you want to batch enrich indicators. This command works on existing indicators only (it does not create them on its own). When running the command, the relevant enrichment command is triggered (such as `!ip`), which is based on the indicator type that is found. The data is saved to context and to the indicator.

#### **NOTE:**

Triggering enrichment on a substantial amount of indicators can take time (since it's activating all enrichment integrations per indicator) and can result in performance degradation.

- Reputation commands (such as `!ip`)

This command can work on existing and non-existing indicators. If extraction is on, the data is saved both to the indicator and the issue's context. If not, then just to the context because the mapping flow is always triggered in enrichment commands. The default configuration is set to none in playbook tasks for extraction.

The indicator does not need to exist to run the reputation command, as the command uses a third-party threat intel integration, such as Unit 42 Intelligence, IPInfo, etc. You can also click the Enrich indicator button in the indicator layout.

#### 4.4.8.7 | Update issue fields

Abstract

Use a playbook, script, or command to update issue fields.

You can update issue fields by running the `setIssue` and `setIssueStatus` commands in the CLI, in a script, or a playbook task.



- **setIssue**: Sets values for specific issue fields. The supported fields are presented in the list of arguments.

Example 65. Examples of the setIssue command in the CLI

The following examples show how to run the **setIssue** command in the CLI. You can run CLI commands in the War Room. When you start typing the CLI provides the available options and if you select an enum field, the CLI provides the available values.

- To change the issue severity to **high**, run

```
!setIssue severity=high
```

- To change the issue severity to **high** and star the issue, run

```
!setIssue severity=high starred=true
```

- **setIssueStatus**: Sets the status or resolution value for an issue. This command supports the **status** argument, which presents a list of status and resolution type values. The selected status is set in the **custom\_status** field.

If you specify a resolution status, the issue is closed and the **resolution\_status** and **closeReason** fields are updated to the same value as the **custom\_status** field. If you specify a New, Reopened, or Under Investigation status, the issue remains open and the **resolution\_status** and **closeReason** fields are empty.

#### TIP:

You can create custom issue statuses and resolution reasons, and use the **setIssueStatus** command to set these custom statuses for issues.

For example, when a user starts investigating an issue, the issue status is automatically changed from New to Under Investigation. In some cases, it is useful to create an interim status, such as Triage. After you create the custom status, the new status will be available for selection. To create a custom status, follow the instructions in [Create custom case statuses and resolution reasons](#).

Example 66. Examples of using the setIssueStatus command in the CLI

The following examples show how to run the **setIssueStatus** command in the CLI. You can run CLI commands in the War Room. When you start typing, the CLI provides the available options and if you select an enum field, the CLI provides the available values.

- To change the issue status to **Resolved - Known Issue**, run

```
!setIssueStatus status="Resolved - Known Issue"
```

- To change the issue status to custom status **Triage**, run

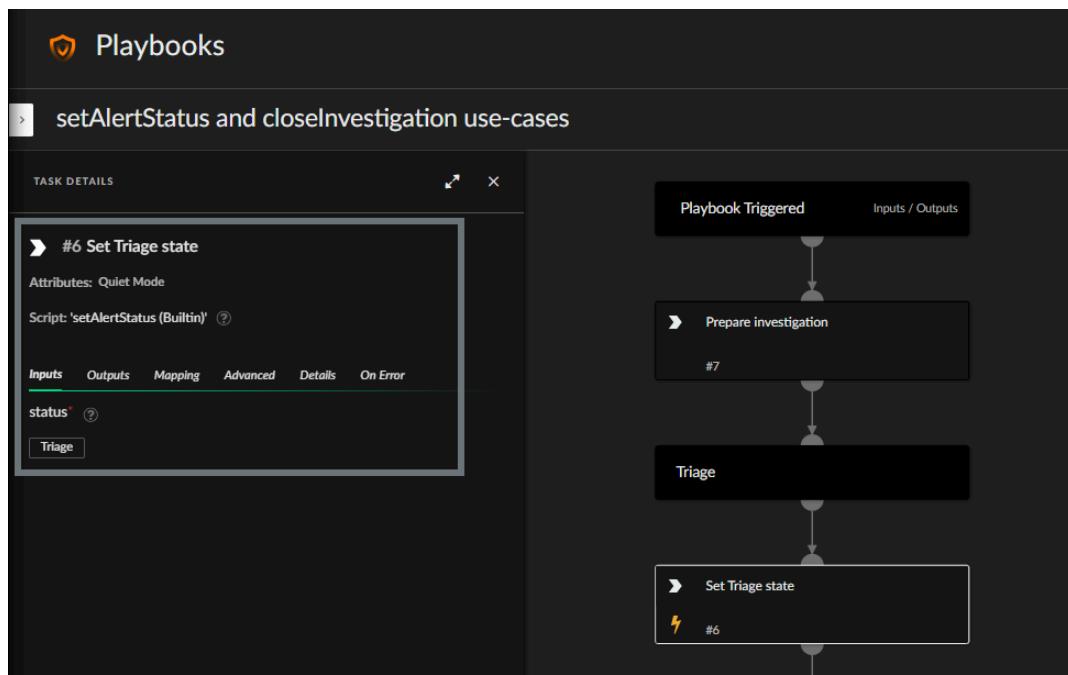
```
!setIssueStatus status=Triage
```

#### NOTE:

You must create a custom status before you can select it.

Example 67. Example of using the setIssueStatus command in a playbook

The following example shows how the **setIssueStatus** command can be used in a playbook task. In this example, the task sets a custom issue status (Triage). The custom issue status was created before setting up the playbook.



#### 4.4.8.8 | Use issue timer field commands manually in the CLI

##### Abstract

Set and change timers for specific issues, such as decreasing the required response time for a high-priority issue.

You can manage the timers for a specific issue by running commands manually in the CLI. By running CLI command you can to manage timers on a more granular level within specific issues when the need arises. For example, for a high severity issue you might need to decrease the response time.

##### Set timer fields

Use the **setIssue** command to set a specific issue due date, or to set a specific timer field in an issue. If you add the **sla** parameter to the command, it sets the time for the issue's due date. If you also add the **slaField** you set the timer for the issue field.

Example 68.

To change the Time to Assignment field target to 30 minutes in the current issue, run the following command:

```
!setIssue sla=30 slaField=timetoassignment
```

To change the timer to February 1, 2024, at 11.12 am, run the following command:

```
!setIssue sla=2024-02-01T11:12
```

##### NOTE:

When defining the values for the **slaField** use the machine name for the field, which is lowercase and without spaces. You can check the machine name by editing the issue field.

##### Start or stop timer fields

Run the following commands in the CLI:

Command	Description
<b>startTimer</b>	<p>Starts the timer.</p> <p>This command can also be used to restart a paused timer.</p> <p>Example 69.</p> <pre>!startTimer timerField=timetoassignment</pre> <p><b>NOTE:</b></p> <p>Timer fields are not started automatically when an issue is created unless run in a playbook.</p>
<b>pauseTimer</b>	<p>Pauses the timer.</p> <p>Use this command when a timer field has already started.</p> <p>Example 70.</p> <pre>!pauseTimer timerField=timetoassignment</pre>
<b>stopTimer</b>	<p>Stops the timer.</p> <p>Example 71.</p> <pre>!stopTimer timerField=timetoassignment</pre> <p><b>NOTE:</b></p> <p>After a timer field is stopped, before you can start the timer again you must reset the timer using the <b>resetTimer</b> command.</p> <p>Timers are automatically stopped when an issue is closed.</p>



Command	Description
<code>resetTimer</code>	<p>Clears all fields for the timer.</p> <p>This command must be used before restarting a timer that was stopped.</p> <p>Example 72.</p> <pre>!resetTimer timerField=timetoassignment</pre>

#### NOTE:

When running commands in the CLI, you can specify the `alertID` to change the timer for a different issue.

#### 4.4.8.9 | Close an issue

##### Abstract

You can close an issue by running the `closeInvestigation` command.

Once you complete your investigation, perform one of the following actions to close an issue:

- Manually close an issue: Right-click an issue and select Change Status â†“ Resolved and select a resolution reason.
- Automatically close an issue: Run the `closeInvestigation` command in the CLI, in a script, or a playbook task. You can configure this command to run as part of a flow when automating issue investigation.

The `closeInvestigation` command supports the `closeReason` and `closeNotes` arguments. The `closeReason` argument accepts a free text value; however, if the free text value doesn't match one of the defined resolution reasons the `resolution_status` field is set to `Resolved - Other`. To see a description of the resolution reasons, see Resolution reasons for cases and issues.

#### NOTE:

When an issue is resolved it remains linked to a case. Once all of the issues in a case are resolved, the case is automatically closed.

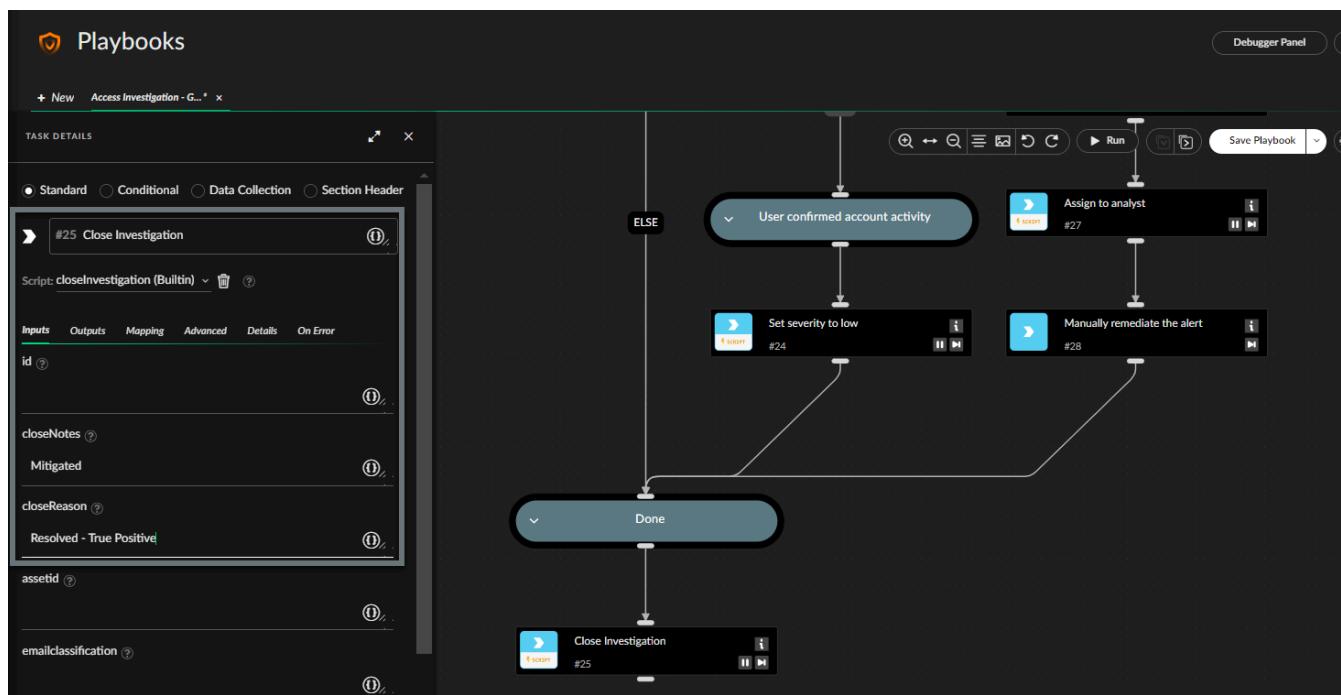
Example 73. Example of using the `closeInvestigation` command in the CLI

In this example, the command specifies to close the issue and set values for `closeReason` and `closeNotes`.

```
!closeInvestigation closeReason="Resolved - Known Issue" closeNotes= "Mitigated"
```

Example 74. Example of using the `closeInvestigation` command in a playbook

In this example, the `closeInvestigation` command is used in a playbook and values are set for `closeReason` and `closeNotes`.



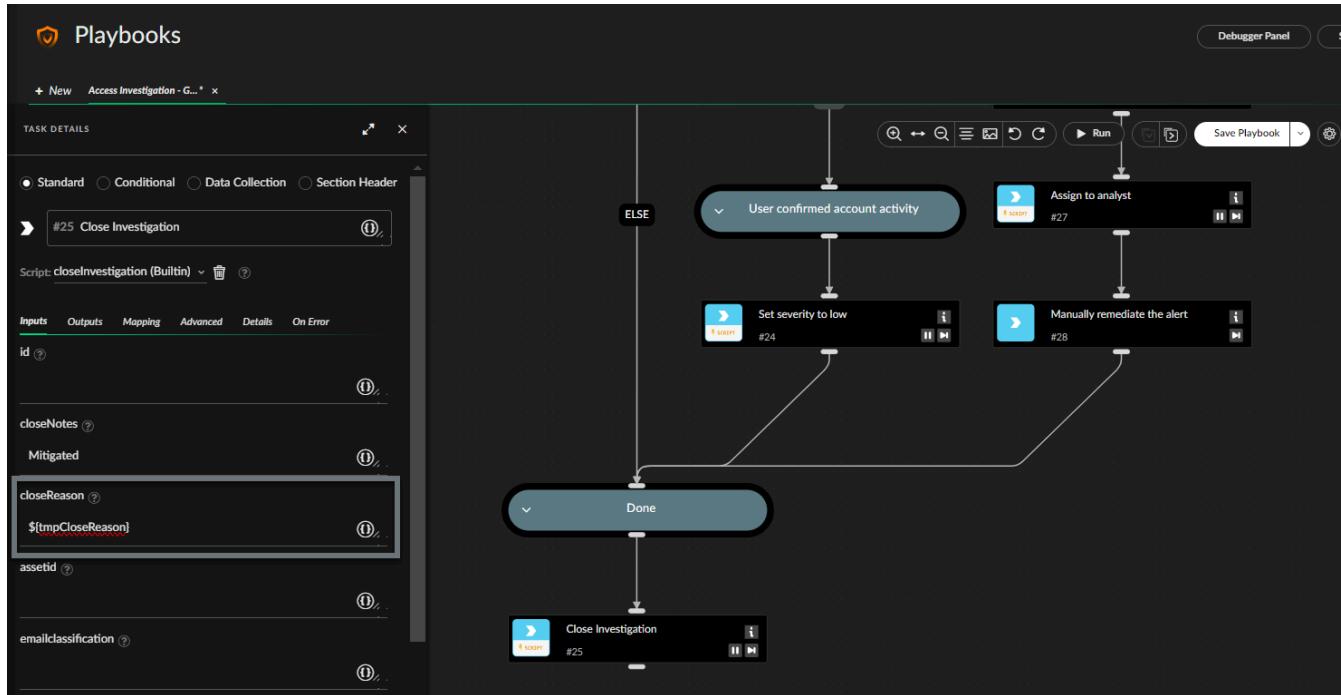
#### Example 75. Example of using a variable in the closeReason field

In this example the close reason field specifies the \${tmpCloseReason} variable value. The tmpCloseReason key was added to the issue context data, and the value is drawn from this field.

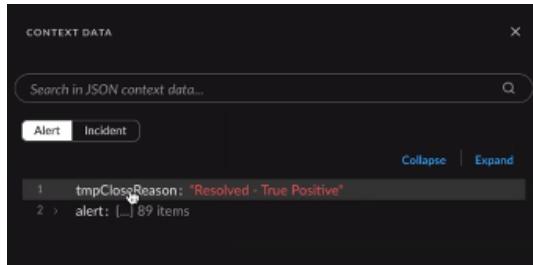
1. Add the tmpCloseReason key and set the value, run the following command in the issue War Room:

```
!Set key=tmpCloseReason value="Resolved - True Positive"
```

2. Create a task in your playbook for the closeInvestigation command and set the closeReason field to \${tmpCloseReason}.



When the playbook runs, it draws the value from this field in the context data:



## 4.5 | Investigate artifacts and assets

### Abstract

You can investigate specific artifacts and assets on dedicated views related to IP address, Network Assets, and File and Process Hash information.

From the Cases view, open the Key Assets & Artifact tab to see the assets and artifacts that are associated with the case, including hosts, IP addresses, and users. Icons represent properties of the artifacts and assets. Hover over an icon for more information. Click the more options icon to drill down in dedicated views, or take actions on the asset or artifact. The Key Assets & Artifact tab shows the following information:

- Artifacts

To aid you with threat investigation, Cortex AgentIX displays the WildFire-issued verdict for each key artifact in a case. To provide additional verification sources, you can integrate external threat intelligence services with Cortex AgentIX.

- Assets

Displays Hosts and Users details. For hosts with a Cortex XDR agent installed, click on the host name to see more information in the Details panel.

### 4.5.1 | Investigate an IP address

### Abstract



Investigate cases, connections, and threat intelligence reports related to a specific IP address on the IP View.

Drill down on an IP address on the IP View. On this view, you can investigate and take actions on IP addresses, and see detailed information about an IP address over a defined 24-hour or 7-day time frame. In addition, to help you determine whether an IP address is malicious, the IP View displays an interactive visual representation of the collected activity for a specific IP address.

How to investigate an IP address

1. Open the IP View.

Right-click the IP address that you want to investigate and select Open IP View.

2. In the left panel, review the overview of the IP address.

The overview displays network operations, cases, actions, and threat intelligence information relating to the selected IP address, and provides a summary of the network operations and processes related to the IP address.

The displayed information and available actions are context-specific.

a. Add an Alias or Comment to the IP address.

b. Review the location of the IP address. By default, Cortex AgentX displays information on whether the IP address is an internal or external IP address.

- Externalâ€ Connection Type: Incoming displaying IP address is located outside of your organization. Displays the country flag if the location information is available.
- Internalâ€ Connection Type: Outgoing displaying IP address is from within your organization. The XDR Agent icon is displayed if the endpoint identified by the IP address had an agent installed at that point in time.

c. Identify the IOC severity.

The color of the IP address value is color-coded to indicate the IOC severity.

d. Review threat intelligence for the IP address.

Depending on the threat intelligence sources that are integrated with Cortex AgentX, the following threat intelligence might be available:

- Virus Total score and report

**NOTE:**

Requires a license key. Select Settings â€“ Configurations â€“ Integrations â€“ Threat Intelligence.

- Whois identification data for the specific IP address.
- IOC Rule, if applicable, includes the IOC Severity, Number of hits, and Source.
- EDL IP address if the IP address was added to an EDL.

e. Review the related cases.

Recent Open Cases lists the most recent cases that contain the IP address as part of the caseâ€ s key artifacts, according to the Last Updated timestamp. If the IP address belongs to an endpoint with a Cortex XDR agent installed, the cases are displayed according to the hostname rather than the IP address. To dive deeper into a specific case, select the case ID.

3. In the right-hand view, use the filter criteria to refine the scope of the IP address information that you want to visualize in the map.

In the Type field, select Host Insights to pivot to the Asset View of the host associated with the IP address, or select Network Connections to display the IP View of the network connections made with the IP address.

4. Review the selected data.

- Select each node for additional information.
- Select Recent Outgoing Connections to view the most recent connections made by the IP address. Search all Outgoing Connections to run a Network Connections query on all the connections made by the IP address.

5. Perform actions on IOC or EDL.

Depending on the current IOC and EDL status, the Actions button is displayed.

#### 4.5.2 | Investigate a file and process hash

Abstract

Investigate cases, actions, and threat intelligence reports related to a specific file or process hash on the Hash View.



Drilldown on a file or process hash on the Hash View. On this view you can investigate and take actions on SHA256 hash processes and files, and see information about a specific SHA256 hash over a defined 24-hour or 7-day time frame. In addition, you can drill down on each of the process executions, file operations, cases, actions, and threat intelligence reports relating to the hash.

#### How to investigate a file or process hash

##### 1. Open the Hash View.

Identify the file or process hash that you want to investigate and select Open Hash View.

##### 2. In the left panel, review the overview of the hash.

###### a. Review the signature of the hash, if available.

###### b. Identify the WildFire verdict.

The color of the hash value is color-coded to indicate the WildFire report verdict:

###### WildFire color key

- Blue → Benign
- Yellow → Grayware
- Red → Malware
- Light gray → Unknown verdict
- Dark gray → The verdict is inconclusive

###### c. Add an Alias or Comment to the hash value.

###### d. Review threat intelligence for the hash.

Depending on the threat intelligence sources that are integrated with Cortex AgentIX, the following threat intelligence might be available:

- Virus Total score and report.

###### NOTE:

Requires a license key. Go to Settings → Configurations → Integrations → Threat Intelligence.

- IOC Rule, if applicable, including the IOC Severity, Number of hits, and Source according to the color-coded values:
- WildFire analysis report.

###### e. Review if the hash has been added to:

- Allow List or Block List.
- Quarantined, select the number of endpoints to open the Quarantine Details view.

###### f. Review the recent open cases that contain the hash as part of the case's Key Artifacts according to the Last Updated timestamp. To dive deeper into specific cases, select the Case ID.

##### 3. In the right hand view, use the filter criteria to refine the scope of the IP address information that you want to visualize.

###### Filter criteria

Filter	Description
Event Type	Main set of values that you want to display. The values depend on the selected type of process or file.
Primary	Set of values that you want to apply as the primary set of aggregations. Values depend on the selected Event Type.
Secondary	Set of values that you want to apply as the secondary set of aggregations.
Showing	Number of Primary and Secondary aggregated values to display.



Filter	Description
Timeframe	Time period over which to display your defined set of values.

4. Review the selected data.

To view the most recent processes executed by the hash, select Recent Process Executions. To run a query on the hash, select Search all Process Executions.

5. (Optional) Perform actions on the hash.

## 5 | Threat Intel Management

Enhance your investigation with Threat Intel Management (TIM).

### 5.1 | Get started with Threat Intel Management

Abstract

Learn how to use TIM in your investigations

Before diving in, you should understand the Cortex AgentiX Threat Intelligence Management's functionality and how it integrates with your needs. Review the use cases and key details to optimize your Cortex AgentiX experience from the start. Threat Intel management includes the following features:

- Deep dive into an indicator on the AGENTIX Indicators page.
- Manage Indicator Relationships

#### 5.1.1 | What is Threat Intel Management?

Abstract

Why use TIM with use cases.

The Cortex AgentiX native threat intel management capabilities allow you to unify the core components of threat intel, including threat intel aggregation, scoring, and sharing. Cortex AgentiX automates threat intel management by ingesting and processing indicator sources, such as feeds and lists, and exporting the enriched intelligence data to SIEMs, firewalls, and any other system that can benefit from the data. These capabilities enable you to sort through millions of indicators daily and take automated steps to make those indicators actionable.

**NOTE:**

You can do the following:

- Import from integrations (such as third-party feeds and WildFire)
- Create indicators during investigations
- Create issue rules (such as IP, Domain, and File indicators)
- Export indicators through a feed



## Why Threat Intel Management?

- Powerful native centralized threat intel

Supercharge investigations with instant access to a large repository of built-in, high-fidelity Palo Alto Networks threat intelligence crowdsourced from the largest footprint of network, endpoint, and cloud intel sources.

- Indicator relationships

Indicator connections enable structured relationships between threat intelligence sources and issues. These relationships surface important context for security analysts on new threat actors and attack techniques.

- Hands-free automated playbooks with extensible integrations

Take automated action to shut down threats across over 600 third-party products with purpose-built playbooks based on proven SOAR capabilities.

- Granular indicator scoring and management

Take charge of your threat intel with playbook-based indicator lifecycle management and transparent scoring that can be easily extended and customized.

- Automated, multi-source feed aggregation

Eliminate manual tasks with automated playbooks to aggregate, parse, prioritize, and distribute relevant indicators in real-time to security controls for continuous protection.

- Most comprehensive marketplace

The largest community of integrations with content packs that are prebuilt bundles of integrations, playbooks, dashboards, field subscription services, and all the dependencies needed to support specific security orchestration use cases.

## Threat Intel with Security orchestration

Security orchestration, automation, and response (SOAR) solutions have been developed to weave threat intelligence management into workflows by combining TIM capabilities with case management, orchestration, and automation capabilities. SOAR solutions weave threat intelligence into a more unified and automated workflow. It matches issues both to their sources and to compiled threat intelligence data and can automatically execute an appropriate response.

As part of the extensible Cortex AgentiX platform, TIM unifies threat intelligence aggregation, scoring, and sharing with playbook-driven automation. It empowers security leaders with instant clarity into high-priority threats to drive the right response across the entire enterprise.

Cortex AgentiX provides a common platform for cases and threat information, where there is no disconnect between external threat data and your environment. Automated data enrichment of indicators provides analysts with relevant threat data to make smarter decisions.

Integrated case management allows for real-time collaboration, boosts operational efficiencies across teams, and automates playbooks to speed response across security use cases.

Cortex AgentiX collects data from sources such as issues, and external threat intel feeds. After the data is ingested, Threat Intel playbooks examine the data proactively. The data gets deduped, normalized, and stored in the Threat Intel database so that a Threat Intel analyst can start a threat analysis. The analyst can then send that information to firewalls, share it with other stakeholders, and take remedial action as necessary.

### 5.1.2 | Threat Intel Management use cases

#### Abstract

Typical use cases for analysts and how to set up the use cases by administrators.

The following examples illustrate typical use cases for Threat Intel Management analysts.

#### Dynamic allow lists for business-critical SaaS apps

In this example, Firewall Admins are responsible for ensuring employees can always access SaaS applications such as Zoom and Office 365. They need to manage a stream of inbound change requests from the security team and other business units. Regardless of these daily changes, critical apps must always be allowed. The network infrastructure of SaaS applications is constantly changing/rotating IP addresses and Domains.

1. Configure a feed integration such as Office 365, Amazon AWS, or Unit 42.

1. Navigate to Settings → Data Sources & Integrations.

2. Search for and select the relevant integration and click Add Instance.

In this example, add the AWS feed.

3. Set up the instance. In the Indicator Reputation field, select Benign.

4. Test and save the instance.



2. (Optional) Configure a playbook to filter indicators according to your requirements.
3. Go to the Indicators page and run the following search to return IP, IPv6 or IPv6CIDR results:

```
sourceBrands:"AWS Feed" and expirationStatus:active and type:IP or type:IPv6 or type:IPv6CIDR
```

4. Configure the Generic Export Indicator Service integration.

1. On the Data Sources & Integrations page, search for and select Generic Export Indicators Service, and click Add Instance.
2. In the Indicator Query field, add the query in step 3.
3. Add the remaining fields, test, and save.

5. Test the EDL by running the cURL command: curl -v-u- user:pass https://ext-<tenant>crtx<region>.paloaltonetworks.com/xsoar/instance/execute/<instance-name>

#### Proactive blocking of known threats

The security team needs to leverage threat intelligence to block or alert on known bad domains, IPs, hashes, etc. (indicators). The indicators are collected from many sources, which need to be normalized, scored, and analyzed before pushing to security devices such as firewalls for alerting. Detection tools can only handle limited amounts of threat intelligence data and need to constantly re-prioritize indicators.

#### Solution

Indicator prioritization. Cortex AgentIX can ingest phishing issues from email inboxes through integrations. Once an issue is ingested, a playbook is triggered and can have any combination of automated or manual actions that users desire. The playbooks can have filters and conditions that execute different branches depending on certain values.

1. Configure feed integrations such as Unit 42 Feed, TAXII feed, etc.

1. Navigate to Settings → Data Sources & Integrations.
2. Search for and select the relevant threat intel feed integration and select Add Instance.
3. Set up the instance.  
Leave the Indicator Reputation field blank.
4. Test and save the instance,

2. (Optional) Configure a playbook to filter indicators according to your requirements.

3. Go to the AGENTIX Indicators page and run the following search to return IP addresses with the verdict malicious with high reliability:

```
expirationStatus:active and type:IP and verdict:malicious and aggregatedReliability:A - Completely reliable
```

4. Configure the Generic Export Indicator Service integration.

1. Navigate to Settings → Data Sources & Integrations.
2. Search for and select Generic Export Indicators Service and click Add Instance.
3. In the Indicator Query field, add the query in step 3.
4. Add the remaining fields, test, and save.

5. Test the EDL by running the cURL command: curl -v-u- user:pass https://ext-<tenant>crtx<region>.paloaltonetworks.com/xsoar/instance/execute/<instance-name>

You can use this URL in your Next-Generation Firewall.

#### 5.1.3 | Roles and responsibilities in Threat Intel Management

##### Abstract

Roles and responsibilities in a Threat Intel Management environment.

A Threat Intel Management (TIM) analyst may have a different persona in the SOC. In some organizations, the TIM analyst is part of the SOC analyst's definition of work, but they have different workflows and use cases. The daily work of SOC analysts and TIM analysts are different.



Roles	Responsibility
Security Analyst (SOC Tier-1)	<ul style="list-style-type: none"> <li>• Triage Specialist</li> <li>• Monitor, manage, and configure security tools</li> <li>• Review cases to assess their urgency</li> <li>• Escalate cases when necessary</li> </ul>
Threat Intel Analyst (SOC Tier 2-3)	<ul style="list-style-type: none"> <li>• Case responders and threat hunters</li> <li>• Remediation of escalated cases from Tier 1 - investigation, response, and assessments</li> <li>• Proactive work to remove infrastructure weaknesses</li> </ul>

#### 5.1.4 | Indicator concepts

Before you start customizing and investigating you should be familiar with the following terms

##### Indicators of Compromise

Indicators of compromise (Indicators) are artifacts that can signal a security breach has occurred and are associated with security issues. They help correlate issues, create hunting operations, and enable you to easily analyze issues and reduce Mean Time to Response (MTTR). They are an essential part of the case management and remediation process. Indicators can include:

- IP addresses: Unusual or foreign IP address accessing your network
- Hashes: Unique identifiers for files or malware
- Domain names: Suspicious or malicious domains
- Registry entries: Changes to the system registry

##### Fetch indicators

Cortex AgentiX includes integrations that fetch indicators from a vendor-specific source, such as TAXII, or a generic source, such as a CSV or JSON file. For more information about how to set up a Threat Intel feed integration to fetch indicators, see [Configure Threat Intelligence feed integrations](#).

##### Indicator ingestion

Cortex AgentiX automates threat intel management by ingesting and processing indicator sources, such as feeds and lists, and exporting the enriched intelligence data to SIEMs, firewalls, and any other system that can benefit from the data. These capabilities enable you to sort through millions of indicators daily and take automated steps to make those indicators actionable in your security posture.

##### NOTE:

You can store up to 100,000,000 indicators.

Indicators are added to Cortex AgentiX via the following methods:

Method	Description	Classification And Mapping
Integration	Feed integrations: Fetch indicators from a feed, for example, TAXII, Office 365, and Unit 42 Feed.	
Indicator extraction		Only the value of an indicator is extracted, so no classification or mapping is needed.



Method	Description	Classification And Mapping
Manual	<ul style="list-style-type: none"> <li>• Command line</li> <li>• Mark: The user marks a piece of data as an indicator.</li> <li>• STIX file: Manually upload a STIX file on the AGENTIX Indicators page.</li> </ul>	<p>Data is inserted manually via the UI so no classification or mapping is needed.</p> <p>If importing a STIX file, mapping is done via the STIX parser code.</p>

#### Common indicator data model

When indicators are ingested, regardless of their source, they have a unified, common set of indicator fields, including traffic light protocol (TLP), expiration, verdict, and tags.

#### Indicator smart merge

The same indicator can originate from multiple sources and be enriched with multiple methods (such as integrations, scripts, and playbooks). Cortex AgentIX implements a smart merge logic to make sure indicators are accurately scored (verdict) and aggregated. Indicator fields are merged according to the source reliability hierarchy. When there are two different values for a single indicator field, the field is populated with the value provided by the source with the highest reliability score. For multi-select and tag fields, new values are appended, rather than replacing the original values.

#### Indicators enrichment cache (Insightcache)

To avoid exceeding API quotas for third-party services, indicators are only updated after the cache expiration period. By default, the cache expires 4,320 minutes (3 days) after an indicator is updated, and cannot be cleared manually. The cache expiration can be set in the indicator type parameters. Indicator enrichment cache expiration only applies to automatic enrichment, triggered by the `enrichIndicators` command, and does not apply when you run reputation commands, such as `!ip, directly`.

#### Indicator timeline

The indicator timeline displays an indicator's complete history, such as the first-seen and last-seen timestamp and changes made to indicator fields.

#### Indicator expiration

When ingesting and processing many indicators daily, it's important to control whether or not they are active or expired and to define how and when indicators are expired. Cortex AgentIX offers multiple options to set indicator expiration. To configure how to expire an indicator, see Configure indicator expiration.

#### Exclusion list

Indicators added to the exclusion list are disregarded by the system and are not created or involved in automated flows such as indicator extraction. For more information, see Delete and exclude indicators.

#### Jobs

Administrators can define a job to trigger a playbook when the specified feed or feeds finish a fetch operation that includes a modification to the list. The modification can be a new indicator, a modified indicator, or a removed indicator. To create a job to process indicators, see Create jobs to process indicators example.

### 5.1.5 | Indicator lifecycle

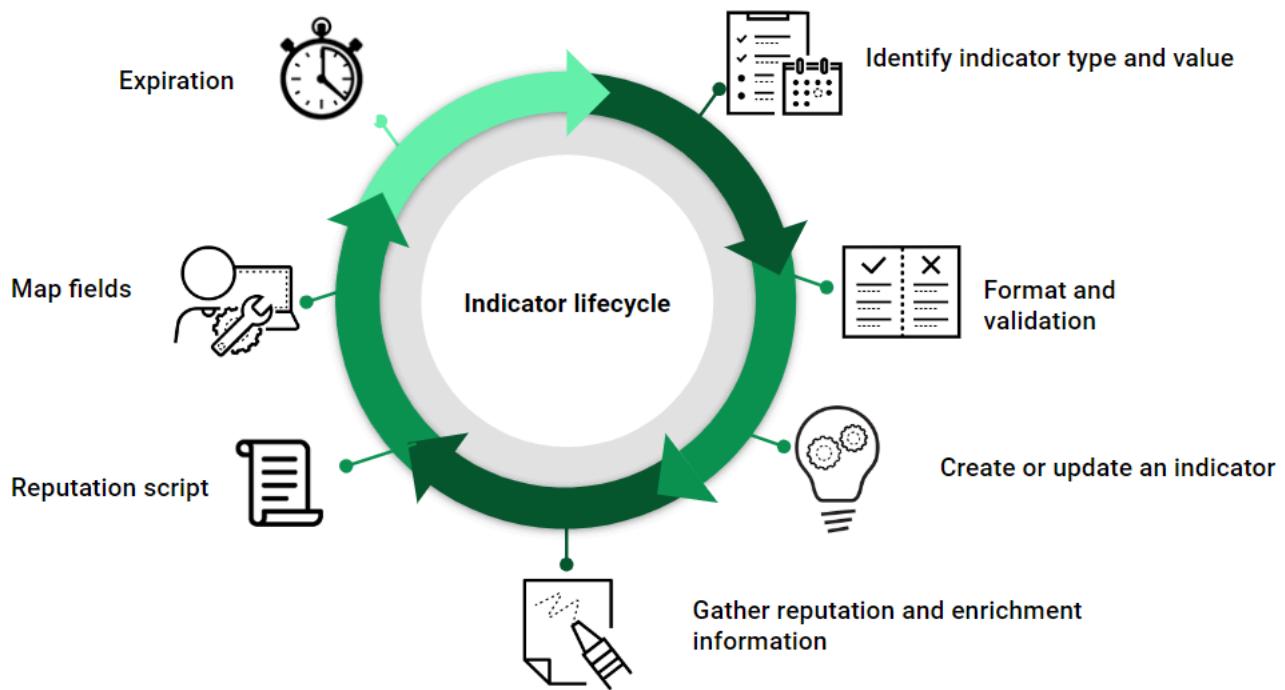
#### Abstract

Indicators are artifacts associated with issues and are an essential part of the case management and remediation process.

Indicators are text-based artifacts associated with issues, such as IP addresses, URLs, and email addresses, and are an essential part of the case management and remediation process. They help correlate issues, create hunting operations, and enable you to easily analyze cases and reduce Mean Time to Response (MTTR).

The following diagram explains the indicator lifecycle in Cortex AgentIX.





Step	Details
1. Identify the indicator type and value	<p>Cortex AgentiX analyzes the text-based artifact and if it matches the indicator type profile. The indicator value is extracted, based on the indicator profile definition. Indicator extraction identifies indicators from various sources within Cortex AgentiX, such as email headers, IP addresses, email addresses, and file hashes in file attachments. For more information about indicator extraction, see <a href="#">Indicator extraction</a>.</p> <p>You can create or customize existing indicator types and fields for your use case. For more information, see <a href="#">Customize indicator fields and types</a>.</p>
2. Formatting and validation	<p>Formatting and validation of the indicator are done using a formatting script that validates the data that represents the indicator's value and determines how we want the data to appear in Cortex AgentiX. For example, the URL indicator type uses the <code>FormatURL</code> script, which defangs URLs. For more information, see <a href="#">Formatting scripts</a>.</p>
3. Create or update an indicator	<p>If the indicator is not known to Cortex AgentiX, an indicator is created or you can create your own. If already known, it is updated with any new data including last seen dates. If the indicator is in an expired state but new data is received, it changes to active status.</p>
4. Gather reputation and enrichment information	<p>You can run reputation commands and enhancement script commands on indicator values. You need to set them to run in the indicator type. The enhancement script also runs on the indicator type. Both determine the indicator's verdict. For more information, see <a href="#">Enhancement scripts</a>.</p> <p>When a reputation command/enhancement script is run, the verdict gets added to the issue context, when attached to an issue. Generally, the information is found under the <code>DBot Score</code> key, the specific Indicator type, and specific vendor information.</p> <p><b>NOTE:</b></p> <p>To run enhancement scripts and reputation commands, you must configure a relevant enrichment integration, such as VirusTotal, IPInfo v2, etc.</p>
5. Reputation scripts	<p>Reputation scripts can be used if you want to override existing reputation commands with custom logic. For those indicator types without reputation commands, a custom reputation script can be applied. Use it to customize verdicts and DBotScore context entry. For more information, see <a href="#">Reputation scripts</a>.</p>



Step	Details
6. Map indicator fields	After your indicator is enriched, you can map fields. Some indicator fields are automatically mapped by Cortex AgentiX to contain the relevant values. The default settings can be changed for each indicator type. You can create and associate any custom fields with indicators. For more information, see <a href="#">Indicator classification and mapping</a> .
7. Expiration	<p>Many indicators have expiration dates as threats are dynamic. IP addresses may change, systems may be fixed, etc. When configuring an indicator type, you can set it never to expire or after a time interval. For more information, see <a href="#">Configure indicator expiration</a>.</p> <p><b>TIP:</b></p> <p>We recommend defining your policy for handling expired indicators.</p>

## 5.2 | Indicator configuration

### Abstract

Create indicator types and fields, customize the exclusion list, indicator reputation, and indicator extraction.

Customize your indicators to your specific needs. Edit existing indicator types and fields, add scripts, and configure tailored extraction and expiration settings for optimal insights.

### 5.2.1 | Customize indicator fields and types

#### Abstract

Customize your indicators to your specific needs. Edit existing indicator types and fields, add scripts, and configure tailored extraction and expiration settings for optimal insights.

Cortex AgentiX provides out-of-the-box indicator types and fields. However, your use case may require indicator customization, either by editing existing indicator types and fields or by creating new ones to help investigate and respond to potential security threats specific to your organization.

Custom indicators can provide more accurate and efficient identification of potential cyber security threats. For example, you can customize indicators to monitor and detect unusual activity within your organization's internal network. This can include creating indicators to flag unauthorized access attempts or unusual data transfers, or identifying insider threats or compromised accounts.

Before customizing an indicator, review the ingested indicator and then customize it as needed. After ingesting issues and indicators, check the indicator information associated with your issue. From an issue, review the context data. If there is information in the context data that you don't see in the indicator, map it into indicator fields and display it in the layout.

You can customize the following:

Option	Description
Indicator type	Customize an indicator type by setting the relevant fields, scripts to run, and reputation command for the indicator type. You can create a new indicator type or you can edit an out-of-the-box indicator type. For more information, see <a href="#">Creating new indicator types</a> .
Indicator fields	Custom indicator fields add specific details or attributes to indicators, helping to better classify and understand the nature of potential security threats. You can edit an existing indicator field or create a new one. After creating a new indicator field, map the field to the relevant context data. You can add the field to an indicator type and view it in an indicator layout. For more information, see <a href="#">Create an indicator field</a> and <a href="#">Map custom indicator fields</a> .

#### 5.2.1.1 | Create an indicator type

#### Abstract

In addition to the system-level indicator types, you can create custom indicator types in Cortex AgentiX.

Indicators are categorized by indicator type, which determines the indicator layout and fields that are displayed and which scripts are run on indicators of that type. Cortex AgentiX includes several out-of-the-box indicator types, such as:



- IP Address
  - Domain
  - URL
  - File

For more information about file indicators and how to configure the file hash, see [File indicators](#).

When you create a new indicator type, you define its properties, including whether and how to format the indicator data and how the verdict is calculated.

1. Go to Settings → Configurations → Object Setup → Indicators → Types.

2. Click New.

3. In the Settings tab, add the required indicator profile, such as name and Regex

For more information, see Indicator type profile.

4. In the Custom Fields tab, map the fields, as required.

For more information, see Map custom indicator fields.

Example 76. Create a company email indicator type

The following example describes how to create a new indicator type to manage employee emails, for example for resource management or inside threat investigation.

Create a new indicator type for the employee email addresses which contain the "our\_company.com" company domain.

- 1 Under Settings â Configuration â Object Setup â Indicators â Types â New, in the Settings tab, define the following

- Name: Company email
  - Regex: `.*?@our_company.com` (simplified to capture all the email addresses using the our\_company.com domain).
  - Reputation command: Not relevant for this example, since we don't want any external enrichment.
  - Formatting script: If more formatting is needed, you can use a formatting script to edit the saved value.
  - Reputation script: If needed, you can create a reputation script to affect the DBot score given to the new custom indicator.

2. In the Custom Fields tab, map custom fields for the new indicator type

You can map fields returned using an integration such as Active Directory to obtain more data about the actual user to whom the email belongs. You can also collect data using integrations such as Okta (MFA, SSO), SIEM, and email security. Fields such as Username, Full name, and various groups the user is part of as well as other identifiers are returned to context and mapped into the indicator using the custom fields.

## New Indicator Type

Settings    Custom Fields

Field	Mapping
Action	Choose data path
Actor	Choose data path
Architecture	Choose data path
Assigned role	Choose data path
Assigned user	Choose data path
Associations	Choose data path
Behavior	Choose data path
Blocked	Choose data path
CPE	Choose data path
CVSS Score	Choose data path

Indicator Sample: ⓘ

Load

## **NOTE:**



If you miss mapping any field, you can create additional new indicator fields and either relate them to all indicator types, or relate them only to the new indicator type (recommended).

#### 5.2.1.1.1 | Indicator type profile

##### Abstract

Create or edit an indicator type and configure fields that determine how the system interacts with indicators of that type.

Each indicator type has its own profile that enables Cortex AgentiX to recognize it across the platform. During the indicator extraction flow, the order of execution is regex, formatting script, reputation command, and reputation script. You can update the following fields when updating an indicator type.

Field	Description
Name	A meaningful name for the indicator type.
Reputation script	The output of the reputation script is a verdict score, which is used as the basis for the indicator verdict. Reputation scripts must be tagged reputation to appear in the list for the indicator type. For more information, see Reputation scripts.  The results of reputation scripts do not print to the War Room in the extraction flow.
Formatting script	Modifies how the indicator displays in Cortex AgentiX.  Formatting scripts must be tagged indicator-format to appear in the list for the indicator type. For more information, see Formatting scripts.
Enhancement script	The enhancement script is not part of the indicator extraction flow and is run manually on the indicator type. Examples of enhancement scripts include an enrichment script and a script that runs a search in an SIEM for the indicator.  After indicators are identified, you can go to the Indicator Quick View page, click the Actions button, and run an enhancement script directly on an indicator. For these scripts to be available in the menu, they need the enhancement tag. For more information, see Enhancement scripts.  When you run an enhancement script, it is the equivalent of running the script in the CLI. The script can write to context, return an entry, etc.
Reputation command	Calculates the reputation of indicators of this type. The verdict (reputation) is only associated with the specific indicator value on which it's run (not the indicator type). The command returns the reputation of the indicator value as an entry with entry context and in some cases also returns context values that can be mapped to the indicator type custom fields.  The results of the reputation command do not print to the War Room in the indicator extraction flow. For more information, see Reputation commands.
Regex	The regular expression (regex) to identify indicators for this indicator type.
Layout	Select the indicator layout to use.



Field	Description
Exclude these integrations for the reputation command	<p>Integrations to exclude when calculating the verdict, evaluating, and enriching indicators of this indicator type. Excluding an integration here prevents it from triggering during automated enrichment processes such as indicator extraction, the <code>enrichIndicators</code> command, or the Enrich button. This setting does not prevent the integration from running if you explicitly execute its command (for example, <code>!url</code> or <code>!ip</code>).</p> <p><b>TIP:</b></p> <p>To control which integrations run when explicitly executing a command (and not during enrichment), add the <code>using-brand</code> argument to specify only the integrations you want to use.</p> <p>For example:</p> <pre data-bbox="573 557 1416 586"><code>!url url="https://www.google.com" using-brand="AutoFocus V2,VirusTotal"</code></pre>
Indicator Expiration Method	<p>The method by which to expire indicators of this type. The expiration method that you select is the default expiration method for indicators of this indicator type.</p> <p>The expiration can also be assigned when configuring a feed integration instance, which overrides the default method.</p> <ul data-bbox="589 804 1459 901" style="list-style-type: none"> <li>Never Expire: indicators of this type never expire.</li> <li>Time Interval: indicators of this type expire after the specified number of days or hours. For more information, see Configure indicator expiration.</li> </ul>
Context path for verdict value (Advanced)	<p>When an indicator is extracted, the entry data from the command is mapped to the issue context. This path defines where in context the data is mapped.</p>
Context value of verdict (Advanced)	<p>The value of this field defines the actual data that is mapped to the context path.</p>
Cache expiration in minutes (Advanced)	<p>The amount of time (in minutes) after which the cache for indicators of this type expire. The default is 4,320 minutes (three days). The cache enables you to limit API requests by only updating indicators after a specific time period has passed. The cache cannot be cleared manually.</p> <p><b>NOTE:</b></p> <p>Indicator cache expiration rules only apply to standard enrichment (for example, running the <code>enrichIndicators</code> command). If you run a reputation command, such as <code>!ip</code>, the command executes even if the cache has not expired.</p>

#### 5.2.1.1.2 | File indicators

##### Abstract

You can have a single file indicator for file objects in Cortex AgentIX or each file can have a hash as its own indicator.

Cortex AgentIX uses a single File indicator for file objects. As a result, files that appear with their SHA256 hash and all other hashes associated with the file, (MD5, SHA1, and SSDeep) are listed as properties of the same indicator. In addition, when ingesting an issue through an integration, all file information is presented as one object.

When investigating an issue, in the Indicators section, click a File indicator. You can see additional information for that indicator, including:



- SHA256
- MD5
- SHA1
- SSDeep
- Associated File Names

The `File.Name` values associated with the indicator hash, based on `File` context objects created in Cortex AgentiX (automatically populated).

- Modified

The date and time the `File` indicator was last modified.

- First Seen

The date and time the file was first seen in Cortex AgentiX.

If the file appears in a different issue with a different name and has any of the same hash values, it automatically associates with the original indicator.

#### **NOTE:**

A new `File` indicator only affects new indicators ingested to the Cortex AgentiX platform. Indicators that were already in Cortex AgentiX continue to appear as their respective hash-related indicators.

Configure each file hash to appear as a separate indicator

By default, Cortex AgentiX uses a single file indicator for file objects. As a result, files that appear with their SHA256 hash and all other hashes associated with the file, (MD5, SHA1, and SSDeep) are listed as properties of the same indicator. In addition, when ingesting an issue through an integration, all file information is presented as one object.

If the file appears in a different issue with a different name, and has any of the same hash values, it automatically associates with the original indicator.

If you want to have each file hash appear as its own indicator, do the following:

1. Go to Settings → Configurations → Object Setup → Indicators → Types.
2. Select the `File` indicator and click **Disable**.
3. Select the following required hashes:
  - File SHA-256
  - File SHA-1
  - File MD5
  - SSDeep
4. Click **Enable**.

The file indicator merging method

When a file is created in the system, whether from a feed, indicator extraction or manually added, its original value is created as the indicator's value, while its complementing hashes are saved as fields.

For example, if a SHA256 indicator is extracted from an email and enriched, an indicator with the SHA256 hash as the value will be created, and any other hash that is found in the enrichment phase (such as MD5, SHA1) will be added as a field. If in the future a file indicator with the same MD5 is created in the system, Cortex AgentiX automatically identifies it and merges the two indicators together into one.

For example, the executable `cmd.exe`'s SHA256 `FF79D3C4A0B7EB191783C323AB8363EBD1FD10BE58D8BCC96B07067743CA81D5` was found in an issue and extracted. It also went through enrichment, which provided the information that the file's MD5 is `D7AB69FAD18D4A643D84A271DFC0DBDF`.

The file indicator includes:

```
ID: 1
Type: File
Value: FF79D3C4A0B7EB191783C323AB8363EBD1FD10BE58D8BCC96B07067743CA81D5
SHA256: FF79D3C4A0B7EB191783C323AB8363EBD1FD10BE58D8BCC96B07067743CA81D5
MD5: D7AB69FAD18D4A643D84A271DFC0DBDF
```

Afterwards, through a custom feed, the `cmd.exe`'s MD5 `D7AB69FAD18D4A643D84A271DFC0DBDF` hash is brought in, and Cortex AgentiX creates an indicator of type `File` with the MD5 hash as its value.

A new file indicator is created:

```
ID: 2
Type: File
```



```
Value: D7AB69FAD18D4A643D84A271DFC0DBDF
MD5: D7AB69FAD18D4A643D84A271DFC0DBDF
```

The automatic merging flow for the File indicator type identifies that the two indicators are the same file and merges them together.

The final file indicator is a consolidation of the two, and is the same as the first example above:

```
ID: 1
Type: File
Value: FF79D3C4A0B7EB191783C323AB8363EBD1FD10BE58D8BCC96B07067743CA81D5
SHA256: FF79D3C4A0B7EB191783C323AB8363EBD1FD10BE58D8BCC96B07067743CA81D5
MD5: D7AB69FAD18D4A643D84A271DFC0DBDF
```

#### 5.2.1.1.3 | Formatting scripts

##### Abstract

Formatting scripts validate input and modify how indicators are displayed.

A formatting script has the following main functions:

- Validate inputs, for example, to check that the top-level domain (TLD) is valid.
- Modify how the indicator appears in Cortex AgentiX such as the War Room.

After indicator values are extracted according to the defined regex, the formatting script can be used to modify how the indicator value appears in the War Room and reports. For example, the IP indicator type uses the `UnEscapeIPs` formatting script, which removes any defanged characters from an IP address, so `127[.]0[.]0[.]1` is formatted to `127.0.0.1`. When you click the IP address in the War Room, you see the formatted IP address. This extracted indicator value is then added to the Threat Intel database.

##### Out-of-the-box Formatting Scripts

You can create a new script, or you can use an out-of-the-box formatting script on the Scripts page, for example:

- `UnEscapeIPs`: Removes escaping characters from IP addresses. For example, `127[.]0[.]0[.]1` transforms to `127.0.0.1`.
- `ExtractDomainAndFQDNFromUrlAndEmail`: Extracts domains and FQDNs from URLs and emails, used by the Domain indicator. It removes prefixes such as proofpoint or safelinks, removes escaped URLs, and extracts the FQDN.
- `ExtractEmailV2`: Verifies that an email address is valid and only returns the address if it is valid.

##### Formatting Script example

In the following example, the `RemoveEmpty` script removes empty items, entries, or nodes from an array.

```
// pack version: 1.2.30
const EMPTY_TOKENS = argToList(args.empty_values);

function toBoolean(value) {
    if (typeof(value) === 'string') {
        if (['yes', 'true'].indexOf(value.toLowerCase()) != -1) {
            return true;
        } else if (['no', 'false'].indexOf(value.toLowerCase()) != -1) {
            return false;
        }
        throw 'Argument does not contain a valid boolean-like value';
    }
    return value ? true : false;
}

function isObject(o) {
    return o instanceof Object && !(o instanceof Array);
}

function isEmpty(v) {
    return (v === undefined) ||
           (v === null) ||
           (typeof(v) == 'string' && (!v || EMPTY_TOKENS.indexOf(v) !== -1)) ||
           (Array.isArray(v) && v.filter(x => !isEmpty(x)).length === 0) ||
           (isObject(v) && Object.keys(v).length === 0);
}

function removeEmptyProperties(obj) {
    Object.keys(obj).forEach(k => {
        var ov = obj[k];
        if (isObject(ov)) {
            removeEmptyProperties(ov);
        } else if (Array.isArray(ov)) {
            ov.forEach(av => isObject(av) && removeEmptyProperties(av));
            obj[k] = ov.filter(x => !isEmpty(x));
        }
        if (isEmpty(ov)) {
            delete obj[k];
        }
    });
}
```



```

    });
}

var vals = Array.isArray(args.value) ? args.value : [args.value];

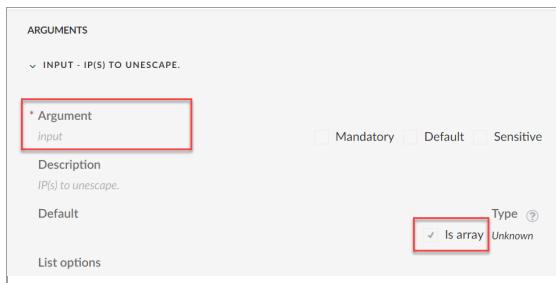
if (toBoolean(args.remove_keys)) {
    vals.forEach(v => isObject(v) && removeEmptyProperties(v));
}
return vals.filter(x => !isEmpty(x));

```

#### Formatting Script input

The formatting script requires a single input argument named `input` that accepts a single indicator value or an array of indicator values. The input argument should be an array to accept multiple inputs and return an entry-result per input.

Argument	Description
<code>input</code>	<p>Accepts a string or array of strings representing the indicator value(s) to be formatted. Will be accessed within the script using <code>demisto.args().get('input', [])</code>.</p> <p>In the script settings, the Is Array checkbox must be selected (see screenshot below). The script code must be able to handle a single indicator value (as string), multiple indicator values in CSV format (as string) and an array of single indicator values (array).</p>



#### Formatting Script outputs

The indicators appear in a human-readable format in Cortex AgentiX. The output should be an array of formatted indicators or an array of entry results (an entry result per indicator to be created). The entry result per input can be a JSON array to create multiple indicators. If the entry result is an empty string, it is ignored and no indicator is created.

Use the `return_results` function to generate the script output. For more information, see [https://xsoar.pan.dev/docs/integrations/code-conventions#return\\_results](https://xsoar.pan.dev/docs/integrations/code-conventions#return_results).

Single-value result:

```

results = CommandResults(
    outputs_prefix='VirusTotal.IP',
    outputs_key_field='Address',
    outputs={
        'Address': '8.8.8.8',
        'ASN': 12345
    }
)
return_results(results)

```

Multiple-value results:

```

results = [
    CommandResults(
        outputs_prefix='VirusTotal.IP',
        outputs_key_field='Address',
        outputs={
            'Address': '8.8.8.8',
            'ASN': 12345
        }
    ),
    CommandResults(
        outputs_prefix='VirusTotal.IP',
        outputs_key_field='Address',
        outputs={
            'Address': '1.1.1.1',
            'ASN': 67890
        }
    )
]
return_results(results)

```

Add a Formatting Script to an indicator type



1. Go to Settings → Configurations → Object Setup → Indicators → Types.

2. Select the indicator type and click Edit.

3. Select the desired formatting script.

**NOTE:**

Formatting scripts must have the **indicator-format** tag to appear in the list.

**NOTE:**

Formatting scripts for out-of-the-box indicator types are system-level, which means that the formatting scripts for these indicator types are not configurable. To create a formatting script for an out-of-the-box indicator type, you need to disable the existing indicator type and create a new (custom) indicator type. If you configured a formatting script before this change and updated your content, this configuration reverts to content settings (empty).

Run a Formatting script in the CLI

You can run out-of-the-box or custom formatting scripts in the CLI to check the extracted indicator data is properly formatted.

The following are examples of the syntax for running the out-of-the-box **UnEscapeIPs** formatting script in the CLI.

- !UnEscapeIPs !UnEscapeIPs input=127.0.0[.]1
- !UnEscapeIPs input=127.0.0[.]1,8.8.8[.]8
- !UnEscapeIPs input=\${contextdata.indicators} (where the key **contextdata.indicators** in the context object is an array)

5.2.1.1.4 | Enhancement scripts

Abstract

Enhancement scripts are run manually and can enrich indicators, write to context, and return entries to the War Room.

Enhancement scripts enable you to gather additional data about the highlighted entry in the War Room. They can enrich indicators, search a SIEM for a specific indicator, write indicator details to context, and return entries to the War Room.

Enhancement scripts are run manually from the Indicator Quick View window or the CLI after indicators are extracted to allow you to collect additional information about an indicator. If you have an issue that contains an IP indicator and you want to run one or more enhancement scripts, go to Indicator Quick View → Actions and under Run Scripts, select the desired script.

**NOTE:**

Enhancement scripts are different from reputation commands. A reputation command runs every integration that has that command within it, to enrich the indicator. The reputation command **ip**, for example, runs every IP integration command in your enabled integrations, to collect data from multiple sources. An enhancement script is manually run after the initial extraction and enrichment for the indicator type is complete.

Enhancement script input

The enhancement script requires the indicator value as the input argument.

Argument	Description
The value of the indicator	For example <b>ip</b> , <b>email</b> , <b>url</b> . The argument name should match the indicator type in lower case. For example, the <b>IPReputation</b> script requires the <b>ip</b> input. For an <b>EmailReputation</b> script the input is <b>email</b> .

In the following example, the **DomainReputation** script uses **domain** as the input.

The screenshot shows a configuration dialog for a script. At the top, it says "DOMAIN - DOMAIN TO LOOK UP". Below that, there's a section for "Argument" with a field labeled "domain" and a checkmark indicating it's "Mandatory". There are also "Default" and "Sensitive" checkboxes. Under "Description", there's a text area with the placeholder "Domain to look up". At the bottom, there's a "Type" dropdown with options "Is array" and "Unknown", currently set to "Unknown".



## Enhancement script outputs

The enhancement script output depends on its input because the script is run manually. If you want the output to be added to indicator enrichment or the Threat Intelligence screen, it should follow the DBotScore convention in the content output as described in <https://xsoar.pan.dev/docs/integrations/dbot>.

```
output =
{
    'Type': entryTypes['note'],
    'ContentsFormat': formats['json'],
    'Contents': "this is the enrichment data",
    'EntryContext': {
        'Email': "xsoar@test.com",
        'DBotScore': {}
    }
}

return_results(output)
```

Add an enhancement script to an indicator type

1. Go to Settings → Configurations → Object Setup → Indicators → Types
2. Select the indicator type and click Edit.
3. Select one or more desired enhancement scripts.

### NOTE:

Enhancement scripts must have the **enhancement** tag applied to appear in the list.

## Run an enhancement script in the CLI

You can run out-of-the-box or custom enhancement scripts in the CLI to enrich specific indicator values.

The following are examples of the syntax for running the out-of-the-box **IPReputation** and **URLReputation** enhancement scripts in the CLI.

- !IPReputation ip=8.8.8.8
- !URLReputation url=cardcom.com

## 5.2.1.1.5 | Reputation scripts

### Abstract

Reputation scripts for indicator enrichment.

Reputation scripts are used to assess and assign reputation scores to indicators. These scripts integrate external threat intelligence or internal data sources to evaluate the reputation of indicators (such as IP addresses, URLs, or file hashes). Reputation scripts enable you to implement custom logic and algorithms for determining the reputation of indicators.

Reputation scripts return the verdict of an indicator as a number. The number overrides the verdict returned from the reputation command and any default settings for the indicator that relates to the verdict, but does not override a manually set verdict.

The system automatically executes the reputation script in the following cases:

- During enrichment: When enrichment is triggered (via indicator extraction, the **enrichIndicators** command, or the Enrich button), the system runs the reputation command and then the reputation script for the specific indicator type.
- If a verdict changes not via the enrichment process: When explicitly running a reputation command such as **!file**, if the result changes the indicator's verdict the reputation script runs to finalize the decision. This happens even if you use the **using** argument to target a specific integration.

The reliability of the score from a reputation script is by default **A++ - Reputation script**.

### Out-of-the-box reputation scripts

You can create a new reputation script, or you can use an out-of-the-box reputation script in the Scripts page, for example:

- CertificateReputation
- cveReputation
- MaliciousRatioReputation
- SSDeepReputation

### Reputation Script input

The reputation requires a single input argument named **input** that accepts an indicator value.



Argument	Description
input	The indicator value.

ARGUMENTS

✓ INPUT - VALUE OF THE INDICATOR.

\* Argument  Mandatory  Default  Sensitive

Description *Value of the indicator.*

Default  Type   Is array

List options

#### Reputation Script outputs

Either a number or a dbotScore. It can either be a raw number which is the score, or a full entry with DBotScore.

```
from CommonServerPython import *
```

```
def main():
    url_list = argToList(demisto.args().get('input'))
    entry_list = []

    for url in url_list:
        entry_list.append({
            'Type': entryTypes['note'],
            'ContentsFormat': formats['json'],
            'Contents': 2,
            'EntryContext': {
                'DBotScore': {
                    'Indicator': url,
                    'Type': 'Onion URL',
                    'Score': 2, # suspicious
                    'Vendor': 'DBot'
                }
            }
        })
    demisto.results(entry_list)

if __name__ in ('__main__', 'builtin', 'builtins'):
    main()
```

#### Values for Common.DbotScore

Constant	Value
Common.DbotScore.NONE	NONE = 0
Common.DbotScore.GOOD	GOOD = 1
Common.DbotScore.SUSPICIOUS	SUSPICIOUS = 2
Common.DbotScore.BAD	BAD = 3

#### Add a Reputation Script to an indicator type

1. Go to Settings → Configurations → Object Setup → Indicators → Types.

2. Select the indicator type and click Edit.

3. Select the relevant reputation script.

#### NOTE:

Reputation scripts must have the reputation tag applied to appear in the list.



## Run a Reputation Script in the CLI

You can run out-of-the-box or custom reputation scripts in the CLI to set the verdict for a specific indicator.

The following are examples for running the out-of-the-box `CertificateReputation` and `MalicioiusRationReputation` reputation scripts in the CLI.

- `!CertificateReputation input=<value of the indicator>`
- `!MalicioiusRationReputation input=<value of the indicator>`

### 5.2.1.1.6 | Reputation commands

#### Abstract

Reputation commands run based on the indicator type and return a verdict for the indicator.

Reputation commands are built-in or custom commands that use integrations to provide predefined functionalities for obtaining an indicator verdict for specific indicator types. These commands simplify the process of fetching reputation data from external services or threat intelligence feeds without requiring extensive scripting. Reputation commands come with preconfigured parameters and settings for commonly used threat intelligence sources.

You can set an indicator type to run reputation commands. The command returns the verdict of the indicator as an entry with entry context and may also return context values that can be mapped to the custom fields of the indicator.

#### NOTE:

Running a reputation command directly (such as `!ip`) might not apply the result to an indicator, nor does it use the enrichment cache. To ensure an indicator is enriched, and to take advantage of caching, use the `enrichIndicators` command or the Enrich button in the UI. This runs the appropriate reputation command/script based on the indicator type settings. Note that extracted indicators are enriched in the same way.

#### Out-of-the-box reputation commands

You can create a new reputation command, or you can use an out-of-the-box reputation command, for example:

- `ip`
- `file`
- `url`
- `email`
- `domain`

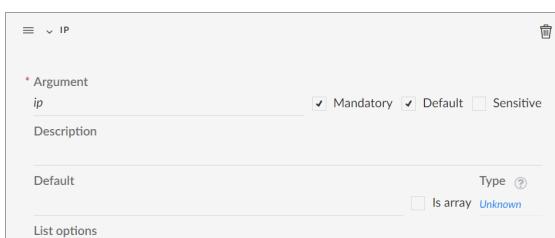
For more details on using out-of-the-box reputation commands or developing new reputation commands, see [Generic Reputation Commands](#).

#### Reputation command input

The reputation command uses the indicator value as the input argument.

Arguments	Description
The value of the indicator	For example <code>ip</code> , <code>email</code> , <code>url</code> . Inputs are based on different integrations. Basic inputs are common to all reputation commands. For example, the <code>!ip</code> command has the following basic inputs:  <code>- name: ip   arguments:     - name: ip     default: true     description: List of IPs.   isArray: true</code>

In this example, the `ip` script uses `ip` as the input, with `is array` unchecked.



## Reputation command output

Outputs return a dbotScore.

Run a Reputation command in the CLI

The following are examples of the syntax for running the `ip`, `domain`, and `file` reputation commands in the CLI.

- `!ip ip=<indicator IP>`
- `!domain domain=<indicator domain>`
- `!file file=<indicator file hash>`
- `!file file=<indicator file hash> using=<a specific integration instance>`

### 5.2.1.7 | Map custom indicator fields

#### Abstract

Learn more about mapping custom indicator fields.

Indicator mapping enables you to automatically update the value of an indicator field without having to manually change it. For example, the IP indicator automatically maps the Country field. If it was not mapped, each time the IP address changes country the analyst would have to update the country every time that indicator type is ingested.

The value of an indicator field is determined by the value of the key in context data the field is mapped to in Cortex AgentiX.

When you start ingesting indicators, the incoming fields are automatically mapped to the relevant indicator fields. Sometimes you may want to change the default settings or map custom indicator fields to specific context data. Before you map custom indicator fields, you need to create the indicator field and add it to the relevant indicator type layout.

#### NOTE:

Some integrations have indicator mappers and classifiers, such as AWS. If you want to use an integration mapper or classifier, see [Indicator classification and mapping](#).

To map custom fields to the indicator type, you need to enrich the indicator either by using the `!enrichindicators` command in the CLI, in a playbook, or by opening an indicator and click Enrich indicator. Enrichment returns an entry, with the `EntryContext` property as the source of the mapping process. When editing an indicator type, in the Custom Fields tab, type the name of the indicator exactly how it appears (in the AGENTIX Indicators page) and click Load.

For the enrichment data to be considered valid, `EntryContext` must include a `DBotScore` with the fields: `Indicator`, `Score`, `Vendor`, and `Type`. If `DBotScore` has those fields, all the data of `EntryContext` is used as the source for the mapping, and not only the data under `EntryContext.DBotScore`.

#### How to map indicator fields

1. Go to Settings → Configurations → Object Setup → Indicators → Types.
2. Select the indicator type and click Edit.
3. Click the Custom Fields tab.

The custom fields associated with this indicator type are listed in the table. If you do not see a custom field in the list, verify that you associated the custom field with this indicator type.

4. (Optional) In the Indicator Sample panel, enter an indicator relevant to the indicator type to load sample data.
5. Click Choose data path to map the custom field to a data path.
  - a. (Optional) Click the curly brackets to map the field to a context path.
  - b. (Optional) From the Indicator Sample panel, select a context key to map to the field.
6. Save the indicator type.

### 5.2.1.2 | Create an indicator field

#### Abstract

Create a new indicator field in the Fields tab in Cortex AgentiX. Add specific indicator information to issue layouts and types.

Indicator fields are used to add specific indicator information to issues. When you create an indicator field, you can associate the field to a specific indicator type or all indicator types.

#### NOTE:



Cortex AgentX IOC fields are based on the STIX 2.1 specifications. For more information, see [Indicator field structure](#).

#### Field types

Field Type	Description
Boolean	Checkbox
Date picker	Adds the date to the field.
Grid (table)	Include an interactive, editable grid as a field type for selected indicator types or all indicator types. When you select Grid (table) you can format the table and determine if the user can add rows.
HTML	Create and view HTML content, which can be used in any type of indicator. <b>NOTE:</b> The following HTML tags are not permitted: <code>blockquote, del, dd, div, dl, dt, fieldset, form, h1, h2, h3, h4, h5, h6, hr, iframe, ins, li, math, noscript, ol, pre, p, script, style, table, ul, address, article, aside, canvas, details, dialog, figcaption, figure, footer, header, hgroup, main, nav, output, progress, section, video</code> . The following CSS tags are not permitted: <code>background-color, text-align, font-size, font-family, font-weight, color, line-height, border-style, border, page-break-inside, tablelayout, padding, background-size, display, padding-top, padding-right, padding-bottom, padding-left, text-size-adjust, break-inside, word-break, width, height, -ms-text-size-adjust, -webkit-text-size-adjust</code> .
Long text	<ul style="list-style-type: none"><li>• Long text is analyzed and tokenized, and entries are indexed as individual words, enabling you to perform advanced searches and use wildcards.</li><li>• Long text fields can't be sorted and used in graphical dashboard widgets.</li><li>• While editing a long text field, pressing Enter will create a new line (case is insensitive).</li></ul> <p>Add a placeholder, if required.</p>
Markdown	Add markdown-formatted text as a template, which will be displayed to users in the field after the indicator is created. Markdown lets you add basic formatting to text to provide a better end-user experience.
Multi select/Array	Select the following options: <ul style="list-style-type: none"><li>• Multi-select from a prefilled (static) list</li><li>• An empty array field for the user to add one or more values as a comma-separated list</li></ul> <p>Add a placeholder, if required.</p>
Number	Can contain any number. Default is 0.
Role	The role assigned to the indicator. Determines which users (by role) can view the indicator.
Short text	<ul style="list-style-type: none"><li>• Short text is treated as a single unit of text and is not indexed by word. Advanced search, including wildcards, is not supported.</li><li>• Short text fields are case-sensitive by default, but can be changed to case-insensitive when creating the field.</li><li>• While editing a short text field, pressing Enter will save the change.</li><li>• Maximum length 60,000 characters</li></ul> <p>Recommended use is one-word entries, such as username and email address.</p> <p>Select a placeholder, if required.</p>



Field Type	Description
Single select	Select a value from a list of options. Add comma-separated values.
Tags	Accepts a single tag or a comma-separated list, not case-sensitive. Add a placeholder, if required.
URL	Add a URL when completing the field.
User	A user in Cortex AgentiX.

#### How to create a field

1. Select Settings → Configurations → Object Setup → Indicators → Fields → New Field.
2. Select the relevant field type.
3. Complete the following fields (if relevant):

Parameter	Description
Mandatory	If selected, this field is mandatory when used in a form.
Field Name	A meaningful display name for the field. After you type a name, you will see below the field that the Machine name is automatically populated. The field's machine name is applicable for searching and the CLI.
Tooltip	An optional tooltip for the field.

4. In the Basic Settings tab, define the values (according to the selected field type).

5. In the Attributes tab define the following:

Field	Description
Script to run when field value changes	The script dynamically changes the field value when script conditions are met. For a script to be available, it must have the <code>field-change-triggered-indicator</code> tag when defining the script. For more information, see Indicator field trigger scripts.
Add to all indicator types	This option is selected by default, which means this field is available to use in all indicator types.  Clear the checkbox to associate this field with a subset of indicator types.

6. Save the field.

7. (Optional) In the indicator type, map custom indicator fields, so an indicator field is automatically updated, without the analyst having to manually change it.

#### 5.2.1.2.1 | Indicator field structure

##### Abstract

Indicator field structure aligned with STIX standards to more easily share and work with IOCs.



Cortex AgentiX IOC fields are based on the STIX 2.1 specifications. These fields provide a guideline for the fields we recommend you maintain within an IOC. None of the fields are mandatory, except the value field. Maintaining this field structure enables you to share and export IOCs to additional threat intel based systems as well as to other cybersecurity devices.

Like STIX, Cortex AgentiX indicators are divided into two categories, STIX Domain Objects (SDOs) and STIX Cyber-observable Objects (SCOs). The category determines which fields are presented in the layout of that specific IOC. In Cortex AgentiX, all SCOS can be used in a relationship with either SDOs or SCOS.

Each IOC table of fields is separated into three parts:

- System fields - Fields created and managed by Cortex AgentiX.
- Custom core fields - Custom fields shared by all IOCs of the same type (SDO or SCO). Fields may be empty.
- Custom unique fields - Fields unique to a specific type of IOC. If a user associates more fields with the IOC, the additional fields are also treated as unique.

#### STIX Cyber-observable Objects (SCO)

##### Account

Similar to STIX User Account Object, this indicator type represents a user account in various platforms such as operating system, social media accounts, and Active Directory. The value for the object is usually the username for logging in.

System Fields		Description
Value		Defines the indicator on Cortex AgentiX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.
Source Time Stamp		When the object was created in the system.
Modified		When the object was last modified.

Custom Fields - Core		Description
Blocked		A Boolean switch to mark the object as blocked in the user environment.
Community Notes		Comments and free-form notes regarding the indicator.
Description		The description of the object.
STIX ID		The STIX ID for the object in the format of account--<UUID>.
Tags		Tags attached to the object.
Traffic Light Protocol		Red, Amber, Green, or White.

Custom Fields - Unique		Description
Account Type		Specifies the type of the account, comes from account-type-ov by STIX.



<b>Custom Fields - Unique</b>		<b>Description</b>
Creation Date	The date the account was created (not the date the indicator was created).	
Display Name	The display name of the account as it is shown in the UI.	
Groups	The groups the account is a member of.	
User ID	The account's unique ID according to the system it was taken from.	

Domain / DomainGlob

Network domain name, similar to the STIX Domain Name object. The value is the domain address.

<b>System Fields</b>		<b>Description</b>
Value	Defines the indicator on Cortex AgentX. The value is the main key for the object in the system.	
Verdict	Malicious, Suspicious, Benign, or Unknown.	
Expiration	The expiration date of the object.	
Source Time Stamp	When the object was created in the system.	
Modified	When the object was last modified.	

<b>Custom Fields - Core</b>		<b>Description</b>
Blocked	A Boolean switch to mark the object as blocked in the user environment.	
Community Notes	Comments and free form notes regarding the indicator.	
Description	The description of the object.	
STIX ID	The STIX ID for the object in the format of <code>domain--&lt;UUID&gt;</code> .	
Tags	Tags attached to the object.	
Traffic Light Protocol	Red, Amber, Green, or White.	

<b>Custom Fields - Unique</b>		<b>Description</b>
Creation Date	The date the domain was created.	



<b>Custom Fields - Unique</b>		<b>Description</b>
DNS Records		All types of DNS records with a timestamp and their values (GRID).
Expiration Date		The domain expiration date.
Certificates		Any certificates issued for the domain.
WHOIS Records		Any records from WHOIS about the domain (GRID).

#### Email

A single user email address.

<b>System Fields</b>		<b>Description</b>
Value		Defines the indicator on Cortex AgentiX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.
Source Time Stamp		When the object was created in the system.
Modified		When the object was last modified.

<b>Custom Fields - Core</b>		<b>Description</b>
Blocked		A Boolean switch to mark the object as blocked in the user environment.
Community Notes		Comments and free form notes regarding the indicator.
Description		The description of the object.
STIX ID		The STIX ID for the object in the format of <code>email--&lt;UUID&gt;</code> .
Tags		Tags attached to the object.
Traffic Light Protocol		Red, Amber, Green, or White.

<b>Custom Fields - Unique</b>
None

#### File



Represents a single file. For backward compatibility, the indicator has multiple fields for different types of hashes. New hashes, however, should be stored under the Hashes grid field. The file value should be its hash (either MD5, SHA-1, SHA-256, or SHA-512, in that order).

<b>System Fields</b>		<b>Description</b>
Value		Defines the indicator on Cortex AgentIX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.
Source Time Stamp		When the object was created in the system.
Modified		When the object was last modified.

<b>Custom Fields - Core</b>		<b>Description</b>
Blocked		A Boolean switch to mark the object as blocked in the user environment.
Community Notes		Comments and free form notes regarding the indicator.
Description		The description of the object.
STIX ID		The STIX ID for the object in the format of <code>file--&lt;UUID&gt;</code> .
Tags		Tags attached to the object.
Traffic Light Protocol		Red, Amber, Green, or White.

<b>Custom Fields - Unique</b>		<b>Description</b>
Creation Date		The file creation date.
File Extension		The file extension.
Associated File Names		Names the file is associated with.
File Type		The type of the file.
Hashes		Any hashes not specified in a separate field.
imphash		The imphash.



<b>Custom Fields - Unique</b>		<b>Description</b>
MD5		The MD5 hash.
Modified Date		When the file was modified on the origin.
Path		The path to the file.
Quarantined		Was the file quarantined?
SHA1		The SHA1 hash.
SHA256		The SHA256 hash.
SHA512		The SHA512 hash.
Size		The file size.
SSDeep		The SSDeep hash.

IPv4 / IPv6 / CIDR / IPv6CIDR

Represents an IP address and its subnet (CIDR). If no subnet is provided, the address is treated as a single IP (same as a /32 subnet).

<b>System Fields</b>		<b>Description</b>
Value		Defines the indicator on Cortex AgentX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.
Source Time Stamp		When the object was created in the system.
Modified		When the object was last modified.

<b>Custom Fields - Core</b>		<b>Description</b>
Blocked		A Boolean switch to mark the object as blocked in the user environment.
Community Notes		Comments and free form notes regarding the indicator.
Description		The description of the object.



<b>Custom Fields - Core</b>		<b>Description</b>
STIX ID		The STIX ID for the object in the format of <code>type--&lt;UUID&gt;</code> .
Tags		Tags attached to the object.
Traffic Light Protocol		Red, Amber, Green, or White.

<b>Custom Fields - Unique</b>		<b>Description</b>
Geo Country		The country where the object is located.
Geo Location		A set of geographic coordinates for the object.
WHOIS records		Any records from WHOIS about the domain (GRID).

#### URL

Represents the properties of a uniform resource locator.

<b>System Fields</b>		<b>Description</b>
Value		Defines the indicator on Cortex AgentIX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.
Source Time Stamp		When the object was created in the system.
Modified		When the object was last modified.

<b>Custom Fields - Core</b>		<b>Description</b>
Blocked		A Boolean switch to mark the object as blocked in the user environment.
Community Notes		Comments and free form notes regarding the indicator.
Description		The description of the object.
STIX ID		The STIX ID for the object in the format of <code>url--&lt;UUID&gt;</code> .
Tags		Tags attached to the object.



<b>Custom Fields - Core</b>		<b>Description</b>
Traffic Light Protocol		Red, Amber, Green, or White.

<b>Custom Fields - Unique</b>		<b>Description</b>
Certificates		Any certificates issued for the domain.

#### STIX Domain Objects (SDO)

##### Attack Pattern

Attack patterns are a type of TTP (Tactics, Techniques and Procedures) that describe ways adversaries attempt to compromise targets. Attack patterns help categorize attacks, generalize specific attacks to the patterns that they follow, and provide detailed information about how attacks are performed. An example of an attack pattern is spear phishing, where an attacker sends a carefully crafted email message with the intent of getting the target to click a link or open an attachment that delivers malware. Attack patterns can also be more specific, such as spear phishing by a particular threat actor (for example, an email saying the target won a contest).

<b>System Fields</b>		<b>Description</b>
Value		Defines the indicator on Cortex AgentIX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.
Source Time Stamp		When the object was created in the system.
Modified		When the object was last modified.

<b>Custom Fields - Core</b>		<b>Description</b>
Community Notes		Comments and free form notes regarding the indicator.
Description		The description of the object.
STIX ID		The STIX ID for the object in the format of <code>attack-pattern--&lt;UUID&gt;</code> .
Tags		Tags attached to the object.
Traffic Light Protocol		Red, Amber, Green, or White.

<b>Custom Fields - Unique</b>		<b>Description</b>
Kill Chain Phases		The list of kill chain phases this Attack Pattern is used for.



<b>Custom Fields - Unique</b>		<b>Description</b>
External References		List of external references consisting of a source and ID. For example, {source: mitere, id: T1189}

#### Campaign

A campaign is a grouping of adversarial behaviors that describes a set of malicious activities or attacks (sometimes called waves) that occur over a period of time against a specific set of targets. Campaigns usually have well defined objectives and may be part of an intrusion set.

Campaigns are often attributed to an intrusion set and threat actors. The threat actors may reuse known infrastructure from the intrusion set or may set up new infrastructure specifically for conducting that campaign.

Campaigns can be characterized by their objectives and the issues they cause, people or resources they target, and the resources (such as infrastructure, intelligence, and malware, tools) they use.

For example, a campaign can describe a crime syndicate's attack using a specific variant of malware and new C2 servers against the executives of ACME Bank during the summer of 2020 to gain secret information about an upcoming merger with another bank.

<b>System Fields</b>		<b>Description</b>
Value		Defines the indicator on Cortex AgentIX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.
Source Time Stamp		When the object was created in the system.
Modified		When the object was last modified.

<b>Custom Fields - Core</b>		<b>Description</b>
Community Notes		Comments and free form notes regarding the indicator.
Description		The description of the object.
STIX ID		The STIX ID for the object in the format of campaign--<UUID>.
Tags		Tags attached to the object.
Traffic Light Protocol		Red, Amber, Green, or White.

<b>Custom Fields - Unique</b>		<b>Description</b>
Aliases		Alternative names used to identify this campaign.
Objective		The campaign's primary goal, objective, desired outcome, or intended effect.

#### Course of action



A course of action is an action taken either to prevent an attack or to respond to an attack that is in progress. It may describe technical, automatable responses (applying patches, reconfiguring firewalls), but can also describe higher level actions such as employee training or policy changes. For example, a course of action to mitigate a vulnerability could describe applying the patch that fixes it.

System Fields		Description
Value		Defines the indicator on Cortex AgentIX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.
Source Time Stamp		When the object was created in the system.
Modified		When the object was last modified.

Custom Fields - Core		Description
Community Notes		Comments and free form notes regarding the indicator.
Description		The description of the object.
STIX ID		The STIX ID for the object in the format of <code>course-of-action--&lt;UUID&gt;</code> .
Tags		Tags attached to the object.
Traffic Light Protocol		Red, Amber, Green, or White.

Custom Fields - Unique		Description
Action		Reserved to capture structured/automated courses of action.

#### CVE

To preserve backward compatibility, our vulnerability indicator is referred to as CVE, but it is equivalent to the Vulnerability object defined by STIX. Unlike STIX, in TIM the object is identified by its CVE number. A vulnerability is a weakness or defect in the requirements, designs, or implementations of the computational logic (code) found in software and some hardware components (firmware) that can be directly exploited to negatively impact the confidentiality, integrity, or availability of that system.

System Fields		Description
Value		Defines the indicator on Cortex AgentIX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.



<b>System Fields</b>		<b>Description</b>
Source Time Stamp		When the object was created in the system.
Modified		When the object was last modified.

<b>Custom Fields - Core</b>		<b>Description</b>
Community Notes		Comments and free form notes regarding the indicator.
Description		The description of the object.
STIX ID		The STIX ID for the object in the format of <b>vulnerability--&lt;UUID&gt;</b> .
Tags		Tags attached to the object.
Traffic Light Protocol		Red, Amber, Green, or White.

<b>Custom Fields - Unique</b>		<b>Description</b>
CVSS Version		The version of the CVSS scoring system.
CVSS Score		The score given to the CVE.
CVSS Vector		The full CVSS vector.
CVSS Table		All CVSS data by Metric - Value pairs.

#### Infrastructure

The Infrastructure SDO represents a type of TTP and describes any systems, software services and any associated physical or virtual resources that support some purpose (for example, C2 servers used as part of an attack, a device or server that is part of a defense, and database servers targeted by an attack). While elements of an attack can be represented by other SDOs or SCOs, the Infrastructure SDO represents a named group of related data that constitutes the infrastructure.

<b>System Fields</b>		<b>Description</b>
Value		Defines the indicator on Cortex AgentIX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.
Source Time Stamp		When the object was created in the system.



<b>System Fields</b>		<b>Description</b>
Modified		When the object was last modified.

<b>Custom Fields - Core</b>		<b>Description</b>
Community Notes		Comments and free form notes regarding the indicator.
Description		The description of the object.
STIX ID		The STIX ID for the object in the format of <code>infrastructure--&lt;UUID&gt;</code> .
Tags		Tags attached to the object.
Traffic Light Protocol		Red, Amber, Green, or White.

<b>Custom Fields - Unique</b>		<b>Description</b>
Aliases		Alternative names used to identify this infrastructure.
Infrastructure types		The type of infrastructure being described. Values should come from STIX <code>infrastructure-type-ov</code> open vocabulary.

#### Intrusion set

An intrusion set is a grouped set of adversarial behaviors and resources with common properties that is believed to be orchestrated by a single organization. An intrusion set may capture multiple campaigns or other activities that are all tied together by shared attributes indicating a commonly known or unknown threat actor. New activity can be attributed to an intrusion set even if the threat actors behind the attack are not known. Threat actors can move from supporting one intrusion set to supporting another, or they may support multiple intrusion sets.

Whereas a campaign is a set of attacks over a period of time against a specific set of targets to achieve an objective, an intrusion set is the entire attack package and may be used over a very long period of time in multiple campaigns to achieve potentially multiple purposes.

<b>System Fields</b>		<b>Description</b>
Value		Defines the indicator on Cortex AgentIX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.
Source Time Stamp		When the object was created in the system.
Modified		When the object was last modified.



Custom Fields - Core		Description
Community Notes	Comments and free form notes regarding the indicator.	
Description	The description of the object.	
STIX ID	The STIX ID for the object in the format of <code>intrusion-set--&lt;UUID&gt;</code> .	
Tags	Tags attached to the object.	
Traffic Light Protocol	Red, Amber, Green, or White.	

Custom Fields - Unique		Description
Aliases	Alternative names used to identify this intrusion set.	
Goals	The high-level goals of this intrusion set, what it is trying to do.	
Primary Motivation	The primary reason, motivation, or purpose behind this intrusion set. Values should come from STIX <code>attack-motivation-ov</code> open vocabulary.	
Secondary Motivation	The secondary reason, motivation, or purpose behind this intrusion set. Values should come from STIX <code>attack-motivation-ov</code> open vocabulary.	
Resource level	Specifies the organizational level at which this intrusion set typically works. Values should come from STIX <code>attack-resource-level-ov</code> open vocabulary.	

#### Malware

Malware is a type of TTP that represents malicious code. It generally refers to a program that is inserted into a system, usually covertly. The intent is to compromise the confidentiality, integrity, or availability of the victim's data, applications, or operating system (OS) or otherwise annoy or disrupt the victim.

System Fields		Description
Value	Defines the indicator on Cortex AgentIX. The value is the main key for the object in the system.	
Verdict	Malicious, Suspicious, Benign, or Unknown.	
Expiration	The expiration date of the object.	
Source Time Stamp	When the object was created in the system.	
Modified	When the object was last modified.	



Custom Fields - Core		Description
Community Notes	Comments and free form notes regarding the indicator.	
Description	The description of the object.	
STIX ID	The STIX ID for the object in the format of <code>malware--&lt;UUID&gt;</code> .	
Tags	Tags attached to the object.	
Traffic Light Protocol	Red, Amber, Green, or White.	

Custom Fields - Unique		Description
Aliases	A list of other names the malware is known as.	
Architecture	The processor architectures (for example, x86, ARM) that the malware instance or family is executable on. The values should come from the STIX <code>processor-architecture-ov</code> open vocabulary.	
Capabilities	Any of the capabilities identified for the malware instance or family. The values should come from STIX <code>malware-capabilities-ov</code> open vocabulary.	
Implementation Languages	The programming language(s) used to implement the malware instance or family. The values should come from the STIX <code>implementation-language-ov</code> open vocabulary.	
Is Malware Family	Whether the object represents a malware family (if true) or a malware instance (if false).	
Malware Types	Which type of malware. Values should come from STIX <code>malware-type-ov</code> open vocabulary.	
Operating System Refs	Identifier of a software object.	

#### Report

Reports are collections of threat intelligence focused on one or more topics, such as a description of a threat actor, malware, or attack technique, including context and related details. They are used to group related threat intelligence together so that it can be published as a comprehensive cyber threat story.

System Fields		Description
Value	Defines the indicator on Cortex AgentIX. The value is the main key for the object in the system.	
Verdict	Malicious, Suspicious, Benign, or Unknown.	
Expiration	The expiration date of the object.	
Source Time Stamp	When the object was created in the system.	



<b>System Fields</b>		<b>Description</b>
Modified		When the object was last modified.

<b>Custom Fields - Core</b>		<b>Description</b>
Community Notes		Comments and free form notes regarding the indicator.
Description		The description of the object.
STIX ID		The STIX ID for the object in the format of <code>report--&lt;UUID&gt;</code> .
Tags		Tags attached to the object.
Traffic Light Protocol		Red, Amber, Green, or White.

<b>Custom Fields - Unique</b>		<b>Description</b>
Publications		Links to publications of the report.

#### Threat actor

Threat actors are individuals, groups, or organizations believed to be operating with malicious intent. A threat actor is not an intrusion set but may support or be affiliated with various intrusion sets, groups, or organizations over time.

<b>System Fields</b>		<b>Description</b>
Value		Defines the indicator on Cortex AgentIX. The value is the main key for the object in the system.
Verdict		Malicious, Suspicious, Benign, or Unknown.
Expiration		The expiration date of the object.
Source Time Stamp		When the object was created in the system.
Modified		When the object was last modified.

<b>Custom Fields - Core</b>		<b>Description</b>
Community Notes		Comments and free form notes regarding the indicator.
Description		The description of the object.
STIX ID		The STIX ID for the object in the format of <code>threat-actor--&lt;UUID&gt;</code> .



Custom Fields - Core		Description
Tags	Tags attached to the object.	
Traffic Light Protocol	Red, Amber, Green, or White.	

Custom Fields - Unique		Description
Alias	A list of other names the threat actor is known as.	
Geo country	The country the threat actor is associated with.	
Goals	The high-level goals of this threat actor, what it is trying to do.	
Resource Level	The organizational level at which this threat actor typically works. Values for this property should come from STIX <code>attack-resource-level-ov</code> open vocabulary.	
Primary Motivation	The primary reason, motivation, or purpose behind this threat actor. Values for this property should come from STIX <code>attack-motivation-ov</code> open vocabulary.	
Secondary Motivation	The secondary reasons, motivations, or purposes behind this threat actor. Values for this property should come from STIX <code>attack-motivation-ov</code> open vocabulary.	
Sophistication	The skill, specific knowledge, special training, or expertise a threat actor must have to perform the attack. Values for this property should come from STIX <code>threat-actor-sophistication-ov</code> open vocabulary.	
Threat actor type	The type(s) of this threat actor. Values should come from STIX <code>threat-actor-type-ov</code> open vocabulary.	

## Tool

Tools are legitimate software used by threat actors to perform attacks. Knowing how and when threat actors use such tools can help understand how campaigns are executed. Unlike malware, these tools or software packages are often found on a system and have legitimate purposes for power users, system administrators, network administrators, or even regular users. Remote access tools such as RDP and network scanning tools such as Nmap are examples of tools that may be used by a threat actor during an attack.

System Fields		Description
Value	Defines the indicator on Cortex AgentiX. The value is the main key for the object in the system.	
Verdict	Malicious, Suspicious, Benign, or Unknown.	
Expiration	The expiration date of the object.	
Source Time Stamp	When the object was created in the system.	
Modified	When the object was last modified.	



Custom Fields - Core		Description
Community Notes	Comments and free form notes regarding the indicator.	
Description	The description of the object.	
STIX ID	The STIX ID for the object in the format of tool--<UUID>.	
Tags	Tags attached to the object.	
Traffic Light Protocol	Red, Amber, Green, or White.	

Custom Fields - Unique		Description
Alias	Alternative names used to identify this tool.	
Tool Types	The kind(s) of tool(s) being described. Values for this property should come from STIX tool-type-ov open vocabulary.	
Tool Version	The version identifier associated with the tool.	
Kill Chain Phases	The list of kill chain phases this attack pattern is used for.	

#### 5.2.1.2.2 | Indicator field trigger scripts

##### Abstract

Associate Cortex AgentIX indicator fields with scripts that are triggered when the field changes.

Indicator field trigger scripts are automated responses that are triggered by a change in an indicator field value. In the script, you define the change in the indicator field value to check for and the actions to take when the change occurs. For example, you can:

- Create a script that runs when the Verdict field of an indicator changes. For example, the script will fetch all issues related to the indicator and take any action that is configured, such as reopening or changing severity.
- Create a script that runs when the Expiration Status field changes. For example, you can define a script that will immediately update the relevant allow/block list and not wait for the next iteration, as seen in the following sample script:

```
indicators = demisto.args().get('indicators')
new_value = demisto.args().get('new')

indicator_values = []
for indicator in indicators:
    current_value = indicator.get('value')
    indicator_values.append(current_value)

if new_value == "Expired":
    # update allow/block list regarding expired indicators
else:
    # update allow/block list regarding active indicators
```

##### Indicator field trigger script arguments

Scripts can be created in Python, PowerShell, or JavaScript on the Scripts page. To use a field trigger script, you need to add the field-change-triggered-indicator tag when creating the script. You can then add the script in the Attributes tab when you edit or create a custom indicator field. If you did not add the tag when creating the script, the script will not be available for use.

Indicator field trigger scripts have the following triggered field information available as arguments (args):



Argument	Description
<code>associatedToAll</code>	Whether the field is associated with all or some indicators. Value: <code>true</code> or <code>false</code> .
<code>associatedTypes</code>	An array of the indicator types the field is associated with.
<code>cliName</code>	The name of the field when called from the CLI.
<code>description</code>	The description of the field.
<code>indicators</code>	A list of indicators that have the current change.
<code>isReadOnly</code>	Specifies whether the field is non-editable. Value: <code>true</code> or <code>false</code> .
<code>name</code>	The name of the field.
<code>new</code>	The new value of the field.
<code>old</code>	The old value of the field.
<code>ownerOnly</code>	Specifies that only the creator of the field can edit. Value: <code>true</code> or <code>false</code> .
<code>placeholder</code>	The placeholder text.
<code>required</code>	Specifies whether this is a mandatory field. Value: <code>true</code> or <code>false</code> .
<code>selectValues</code>	If this is a multi-select type field, these are the values the field can take.
<code>system</code>	Whether it is a Cortex AgentiX defined field.
<code>type</code>	The field type.
<code>user</code>	The username of the user who triggered the script.

Indicator field trigger script best practices



- Indicator field trigger scripts react to changes that have already occurred to the field and cannot validate or block changes to the field.
- Indicator field trigger scripts can be configured on the Verdict, Related Incidents, Expiration Status, and Indicator Type fields, as well as any custom indicator fields.
- Indicator field trigger scripts work in all TIM (Threat Intelligence Management) scenarios and workflows, except for feed ingestion.
- Fields that can hold a list (related incidents, multi-select/tag/role type custom fields) will provide an array of the delta. For example, if a multi-select field value has changed from ["a"] to ["a", "b"], the new argument of the script will get a value of ["b"].
- Indicator field trigger scripts run as a batch. This means that if multiple indicators are changed in the same way and are set to trigger the same action, it will happen in one batch.

For example, in the following scenario for a configured indicator field trigger script named `myTriggerScript` on the Verdict indicator field:

- The Threat Intel Library has two existing Malicious indicators: 1.1.1.1 and 2.2.2.2.
- The user runs the following command `!setIndicators indicatorsValues="1.1.1.1,2.2.2.2" verdict=Benign`.
- The `myTriggerScript` script will run just once, with the following parameters:
  - new - "Benign"
  - old - "Malicious"
  - indicators - "[<indicator\_1.1.1.1>,<indicator\_2.2.2.2>]"
- When writing indicator field trigger scripts, avoid scenarios that call the scripts endlessly (for example, a change in field A triggers script X, which changes field B's value, which in turn calls script Y, which changes field A's value).

#### Add an indicator field trigger script to an indicator field

After creating an indicator field trigger script in the Scripts page in Python, PowerShell, or JavaScript, you can then associate it with an indicator field.

1. Go to Settings → Configurations → Object Setup → Indicators → Fields.
2. Select the indicator field and click Edit.
3. In the Attributes tab, under Script to run when field value changes, select the desired indicator field trigger script.

#### **NOTE:**

Indicator field trigger scripts must have the `field-change-triggered-indicator` tag to appear in the list.

## 5.2.2 | Indicator classification and mapping

### Abstract

Learn about the classification and mapping for indicators.

The following table shows methods by which indicators are detected and ingested in Cortex AgentiX and how they are classified and mapped.

Method	Description	Classification And Mapping
Integration	Feed integrations: Fetch indicators from a feed, for example, TAXII, Office 365, and Unit 42.	Indicator classification and mapping is done in the integration code by duplicating the integration and not in the Indicators → Classification & Mapping tab. For more information, see Feed Integrations.  Some integrations come with a classifier and mapper, which you can customize.
Indicator extraction	If you have enabled system-wide indicator extraction, indicators are extracted from all issues in Cortex AgentiX.	Only the value of an indicator is extracted, so no classification or mapping is needed.  For more information, see Indicator extraction.



Method	Description	Classification And Mapping
Manual	<ul style="list-style-type: none"> <li>Command line</li> <li>Mark: The user marks a piece of data as an indicator.</li> <li>STIX file: Manually upload a STIX file on the Threat Intel (Indicators) page.</li> </ul>	<p>Data is inserted manually via the UI so no classification or mapping is needed.</p> <p>If importing an STIX file, mapping is done via the STIX parser code.</p>

#### Classify and map an indicator type for an integration

The indicator classification and mapping feature enables you to take the data that Cortex AgentiX ingests from integrations, and classify and map the data to indicator types and indicator fields. By classifying the data as different indicator types, you can process them with different playbooks suited to their respective requirements.

#### **NOTE:**

When creating a new indicator type, you classify and map the indicator fields in the indicator type settings. For more details, see [Map custom indicator fields](#).

Classification determines the type of indicator that is created for data ingested from a specific integration. You create a classifier and define that classifier in an integration.

You can map the fields from your third-party integration to the fields in your indicator layouts as follows:

- Map your fields to indicator types irrespective of the integration or classifier. This means that you can create a mapping before defining an instance and ingesting indicators. By doing so, when you do define an instance and apply a mapper, the data that comes in is already mapped.
- Create default mapping for all of the fields that are common to all indicator types. You can still overwrite the contents of a field in the specific indicator type.

#### Classify an indicator type

When an integration fetches indicators, it populates the raw JSON object for the indicator. The raw JSON object contains all of the attributes (fields) for an indicator. For example, source, when the event was created, the priority that was designated by the integration, and more. When classifying ingested indicator data, you want to select an attribute (field) that can determine the indicator type.

Use this procedure to create a classifier or duplicate an existing classifier for ingested indicator data.

1. Select Settings â> Configurations â> Object Setup â> Indicators â> Classification & Mapping.

2. Do one of the following:

a. To create a new classifier, select + New â> Indicator Classifier.

b. To edit an existing classifier, select it and click Edit.

If the classifier is installed from a content pack, you need to duplicate and then edit.

3. Under Get data, select from where you want to import the indicator data. You will classify the indicator type based on this information.

#### **NOTE:**

You can optionally skip importing data. Click the pencil on the right of each indicator type on the right pane to enter the value manually.

- Pull from instance: Select an existing integration instance to import indicator data from.
- Upload JSON: Upload a formatted JSON file that includes the fields you want to classify by.

4. Under Fetched data, select from the attributes (fields) in the imported indicator object a field that will serve as the classifier (key) to route to a specific indicator type.

Cortex AgentiX searches through the imported indicator objects for the values for the field you select.

5. Drag the found values from the Unmapped Values column to the relevant indicator type on the right pane.

6. Save the classifier.

7. Apply the indicator classifier to the relevant feed integration.

a. Navigate to Settings â> Data Sources & Integrations.

b. Select an existing integration instance you want to apply the indicator classifier to or create a new integration instance.

c. In the integration instance settings under Classifier, select the classifier you created and click Save.



Mappers enable you to map the information from ingested indicator data to the indicator fields that you have in your system.

Mapping data takes place in two stages:

1. Map all of the fields that are common to all indicators in the default mapping.
2. Map the additional fields that are specific for each indicator type, or overwrite the mapping that you used in the default mapping.

**NOTE:**

In the Classification & Mapping page, the mapping does not indicate for which indicator types they are configured. When creating a mapper, it is best practice to add to the mapper name and the indicator type the mapper is for. For example, Mail Listener - Phishing.

When mapping a list, we recommend you map to a multi-select field. Short text fields do not support lists. If you need to map a list to a short text field, add a transformer in the relevant playbook task to split the data back into a list.

Use this procedure to create a mapper or duplicate an existing mapper to map all of the ingested indicator fields to an indicator layout.

1. Select Settings â Configuration â Indicators â Classification & Mapping.

2. Do one of the following:

- a. To create a new mapper, select + New â Indicator Mapper (incoming).
- b. To edit an existing mapper, select it and click Edit.

If the mapper is installed from a content pack, you need to duplicate and then edit.

3. Under Get data, select from where you want to import the indicator data. You will map the indicator data based on this information.

- Pull from instance: Select an existing integration instance to import indicator data from.
- Upload JSON: Upload a formatted JSON file that includes the fields you want to map.

4. Under Indicator Type, start by mapping out the Common Mapping. This mapping includes the fields that are common to all of the indicator types and saves you time having to define these fields individually in each indicator type.

5. Click the attribute (field) to which you want to map. You can further manipulate the field using filters and transformers.

6. Repeat this process for the other indicator types for which this mapping is relevant.

7. Save the mapper.

8. Apply the indicator mapper to the relevant feed integration.

- a. Navigate to Settings â Data Sources & Integrations.
- b. Select an existing integration instance you want to apply the classifier to or create a new integration instance.
- c. In the integration instance settings under Mapper, select the mapper you created and click Save.

### 5.2.3 | Indicator extraction

Abstract

Indicator extraction extracts indicators from Cortex AgentiX issue fields and enriches them with commands and scripts.

Indicator extraction identifies indicators from different text sources in the system (such as War Room entries and email content), extracts them (usually based on regex), and creates indicators in Cortex AgentiX . After extraction, the indicator can be enriched.

Indicator enrichment takes the extracted indicator and provides detailed information about the indicator, based on enrichment feeds such as VirusTotal and IPInfo.

**NOTE:**

By default, system-wide automatic indicator extraction and enrichment is disabled. However, if you migrated from Cortex AgentiX 2.x to Cortex AgentiX 3.x, system-wide automatic indicator extraction and enrichment is enabled.

If you have a Threat Intel Management (TIM) Add-on, you can enable or disable automatic indicator extraction system-wide. Go to Settings â Configuration â General â Server Settings. In the Indicators section, enable Enable automatic indicator extraction and enrichment from issues.

To extract indicators from incoming feeds without enrichment or to prevent enrichment for existing indicators, see Exclude indicators from enrichment.

In Cortex AgentiX, the indicator extraction feature extracts indicators from War Room entries and enriches them using commands and scripts defined for the indicator type.

You can extract indicators in the following scenarios:



- When fetching issues
- In a playbook task
- Using the command line

**NOTE:**

Reputation commands, such as `!ip` and `!domain`, can only be used after you configure and enable a reputation integration instance, such as VirusTotal and Whois.

Indicator extraction modes

You set the indicator extraction mode:

- In a playbook task.
- Running a command during an investigation.

Indicator extraction supports the following modes:

- **None:** Indicators are not extracted automatically. Use this option when you do not want to further evaluate the indicators.
- **Inline:** Indicators are extracted within the context that indicator extraction runs (synchronously). The findings are added to the context data. For example, if indicator extraction mode for a task in a playbook is inline, the extraction and enrichment must complete before the next task begins. This option provides the most robust information available per indicator.

**NOTE:**

The inline configuration may delay playbook execution.

**NOTE:**

While indicator creation is asynchronous, indicator extraction and enrichment are run synchronously. Data is placed into the issue context and is available via the context for subsequent tasks.

All indicators are automatically extracted and enriched before a playbook is run. For an on-field change, extraction occurs before the next playbook tasks run.

- **Out of band:** Indicators are extracted in parallel (asynchronously) to other actions. The extracted data will be available within the issue, however, it is not available for immediate use in task inputs or outputs since the information is not available in real-time.

**NOTE:**

When using out of band, the extracted indicators do not appear in the context. If you want the extracted indicators to appear select inline.

- If system-wide indicator extraction and enrichment is enabled, indicators are extracted according to the following system defaults:
  - Issue creation - inline
  - Tasks - none, can be overridden on a per task basis
  - CLI - out of band, but can be overridden on a per-command basis

Troubleshoot indicator extraction

If indicators are not extracted, check whether the indicator mode is set to none, and verify the indicator is not in the Exclusion List, as is or as part of a regular expression (regex).

5.2.3.1 | Set the indicator extraction mode for a playbook task

Abstract

Create indicator extraction rules for a playbook task in Cortex AgentiX. Auto extract for a playbook task. Edit task. Use case indicator extraction.

By default, system-wide indicator extraction is disabled. You can set the indicator extraction mode for specific playbook tasks.

1. Select the playbook where you want to add indicator extraction to a task, and click Edit.
2. In the playbook, click a task to open the Edit Task window.
3. Click the Advanced tab.
4. In the indicator extraction drop-down menu, select the mode you want to use.
5. Click OK.



### 5.2.3.2 | Disable indicator extraction for scripts or integrations

#### Abstract

Disable indicator extraction for a specific script or integration in Cortex AgentIX.

By default, system-wide indicator extraction and enrichment is disabled.

If you have the TIM add-on and you have enabled system-wide indicator extraction and enrichment, the procedure below enables you to disable indicator extraction for a specific script or integration.

- To disable indicator extraction for a script, add the `IgnoreAutoExtract` entry with the value of `true`, when returning an entry.

For example:

```
entry = {
    'Type': entryTypes['note'],
    'Contents': {
        'Echo' : demisto.args()['echo']
    },
    'ContentsFormat': formats['json'],
    'ReadableContentsFormat': formats['markdown'],
    'HumanReadable': hr,
    'IgnoreAutoExtract' : True
}
```

- To disable indicator extraction for an integration, add the '`IgnoreAutoExtract`' entry with the value of `true`, when returning an entry.

For example in the ServiceNow integration:

```
entry = {
    'Type': entryTypes['note'],
    'Contents': result,
    'ContentsFormat': formats['json'],
    'ReadableContentsFormat': formats['markdown'],
    'HumanReadable': tableToMarkdown('ServiceNow ticket', hr, headers=headers, removeNull=True),
    'EntryContext': {
        'Ticket(val.ID==obj.ID)': context,
        'ServiceNow.Ticket(val.ID==obj.ID)': context
    },
    'IgnoreAutoExtract': True
}
entries.append(entry)
return entries
```

For more information about command results in Python, see Python code conventions for `CommandResults`.

### 5.2.4 | Configure indicator expiration

#### Abstract

Cortex AgentIX indicators have an active or expired status, which can be set to expire after a specific period or never to expire. Set default expiration method.

Indicators can have the `Expiration Status` field set to Active or Expired, which is determined by the `Expiration` field. When indicators expire, they still exist in Cortex AgentIX, meaning they are still displayed, and you can still search for them. A job that runs daily checks for newly expired indicators and updates the `Expiration Status` field.

When indicators expire, the expiration status and expiration fields are updated. You can use it to take actions based on indicator expiration. For more information, see `Indicator` field trigger scripts.

You can set the default expiration method for indicators either to never expire or to expire after a specific period. The default expiration method is set by the indicator type. For more information, see `Indicator` type profile.

The following table shows the hierarchy by which indicators are expired.



Method	Description
Manual	<p>Manually expire the indicator either in the indicator layout or CLI. This method overrides all other methods.</p> <p><b>NOTE:</b></p> <p>You need to run CLI commands in the Case or Issue War Room.</p> <p>Use the <code>expireIndicators</code> command to change the expiration status to Expired for one or more indicators. This command accepts a comma-separated list of indicator values and supports multiple indicator types. For example, you can set the expiration status for an IP address, domain, and file hash: <code>!expireIndicators value=1.1.1.1,safeurl.com,45356A9DB614ED7161A3B9192E2F318D0AB5AD10</code></p> <p>Use the <code>!setIndicators</code> command to reset the indicators' expiration value. The parameter's value can either be <code>never</code> or a time in ISO 8601 format. For example, <code>2006-01-02T15:04:05Z</code> (for UTC time) or <code>2006-01-02T15:04:05Z07:00</code> (UTC +7 hours).</p> <p>Examples:</p> <ul style="list-style-type: none"> <li>• <code>!setIndicators indicatorsValues=watson.com expiration=Never</code></li> <li>• <code>!setIndicators indicatorsValues=watson.com expiration=2006-01-02T15:04:05Z</code></li> </ul> <p>You can also use these commands in a script, but the user can override this if running a command in the CLI or the indicator layout.</p>
Feed integration	<p>Some integrations support setting the expiration method on an integration instance level, which overrides the method defined for the indicator type.</p> <p><b>NOTE:</b></p> <p>If a feed's expiration method is set to When removed from the feed, indicators that are removed from the feed immediately expire. Note that if the feed is disabled, its expiration method reverts to that of the indicator type (time-based).</p> <p>Time-based expiration is set according to feed reliability. If the same indicator appears on multiple feeds, the feed with the highest reliability determines the indicator's expiration time. If multiple feeds have the same reliability, the last feed to add or modify the indicator determines its expiration time.</p> <p>Example:</p> <ul style="list-style-type: none"> <li>• An indicator was initially fetched by Feed A, then by Feed B.</li> <li>• Both feeds have the same reliability.</li> <li>• Feed B's indicators are set to expire When removed from the feed.</li> <li>• Feed B is now disabled.</li> </ul> <p>After Feed B is disabled, the indicator's expiration method reverts to that of the indicator type (for example, expire after 7 days). However, if Feed A then modifies the indicator (or removes and re-adds it), the expiration method changes back to Feed A's settings.</p>
Indicator type	<p>The expiration method (interval or never) is defined according to indicator type, which applies to all indicators of this type. This is the default expiration method for an indicator.</p>

## 5.2.5 | Configure Threat Intelligence feed integrations

You can download and install content packs, including threat intelligence integrations such as:

- MITRE ATT&CK
- Unit 42 Feed
- Unit 42 Intelligence
- AlienVault
- AWS

**NOTE:**



Some third-party services (such as Whois and VirusTotal) enforce strict rate limits based on the source IP address. If you encounter quota management issues, we recommend running them on an engine. This routes the traffic through your own private network, ensuring the external service sees a unique, dedicated IP address exclusive to your organization.

How to configure threat intelligence feed integrations

1. Go to Marketplace and install the relevant threat intelligence content pack.
2. Configure the threat intelligence integration by going to Settings → Configurations → Data Collection → Automation & Feed Integrations, search for your integration, and click Add Instance.

The following table is a non-exhaustive list of the most common feed integration parameters. Each feed integration may have parameters unique to that integration. Read the documentation for specific feed integrations for more details.

Parameter	Description
Fetches indicators	Select this option for the integration instance to fetch indicators.  Some integrations can fetch indicators or issues. Select the relevant option for what you need to fetch in the instance.
URL	The URL of the feed.
Feed Fetch Interval	When the integration instance should fetch indicators from the feed.
Indicator verdict	The indicator verdict that will apply to all indicators fetched from this integration instance. See Indicator verdict.
Source reliability	The reliability of the source that provides the threat intelligence data.
Indicator Expiration Method	<p>The method by which to expire indicators from this integration instance. The default expiration method is the interval configured for the indicator type to which this indicator belongs.</p> <ul style="list-style-type: none"><li>• Indicator Type: The expiration method defined for the indicator type to which this indicator belongs (interval or never).</li><li>• Time Interval: Expires indicators from this instance after the specified time interval, in days or hours.</li><li>• Never Expire: Indicators from this instance never expire.</li><li>• When removed from the feed: When the indicators are removed from the feed they are expired in the system.</li></ul> <p><b>NOTE:</b></p> <p>Some feeds only provide information about new indicators and do not specify when indicators are removed. Indicators from these feeds cannot be automatically expired on removal.</p> <p>If a feed's expiration method is set to When removed from the feed, indicators that are removed from the feed immediately expire. Note that if the feed is disabled, its expiration method reverts to that of the indicator type (time-based).</p> <p>Time-based expiration is set according to feed reliability. If the same indicator appears on multiple feeds, the feed with the highest reliability determines the indicator's expiration time. If multiple feeds have the same reliability, the last feed to add or modify the indicator determines its expiration time.</p> <p>Example:</p> <ul style="list-style-type: none"><li>◦ An indicator was initially fetched by Feed A, then by Feed B.</li><li>◦ Both feeds have the same reliability.</li><li>◦ Feed B's indicators are set to expire When removed from the feed.</li><li>◦ Feed B is now disabled.</li></ul> <p>After Feed B is disabled, the indicator's expiration method reverts to that of the indicator type (for example, expire after 7 days). However, if Feed A then modifies the indicator (or removes and re-adds it), the expiration method changes back to Feed A's settings.</p>



Parameter	Description
Bypass exclusion list	When selected, the exclusion list is ignored for indicators from this feed. This means that if an indicator from this feed is on the exclusion list, the indicator might still be added to the system.
Trust any certificate (not secure)	When selected, certificates are not checked.
Use system proxy settings	Runs the integration instance using the proxy server (HTTP or HTTPS) when an engine is selected.
Do not use in CLI by default	Excludes this integration instance when running a generic command that uses all available integrations.

### 5.2.6 | Exclude indicators from enrichment

#### Abstract

Extract and save indicators but do not enrich them.

You can disable enrichment for individual indicators or disable enrichment for all indicators fetched by any of the following feeds:

- Azure Feed
- Office 365 Feed
- Cisco WebEx Feed
- Cloudflare Feed
- Fastly Feed
- AWS Feed
- Zoom Feed
- Public DNS Feed
- Google IP Ranges Feed

If you disable enrichment for an incoming feed, the indicators are extracted and saved but not enriched by Cortex AgentiX, enabling you to conserve system resources when dealing with known indicators.

When an indicator has enrichment excluded, the Enrich Indicator button is disabled. If you try to enrich an indicator that is enrichment excluded, an error will occur.

Indicators of the following indicator types can have enrichment excluded:

- IP
- Domain
- Email
- URL
- File

#### Exclude enrichment for a feed integration

To exclude enrichment for indicators fetched from a feed integration, when configuring an instance of the feed integration, select the Enrichment Excluded checkbox.

#### Exclude enrichment for individual indicators

When creating or editing an indicator of one of the following types: IP, Domain, Email, URL, or File, you have the option to set Enrichment Excluded to Yes or No. The default is No.



## [View list of enrichment excluded indicators](#)

To view the enrichment excluded indicators in the Indicators table, add the Enrichment Excluded column to the table.

### 5.2.7 | Generate issues from indicator rules

#### Abstract

Create issues from indicator rules in Cortex AgentiX.

Indicator rules enable you to generate issues from indicators in Cortex AgentiX for detection. These rules allow you to select indicators or indicator traits to be detected by the tenant. Indicator rules marked for detection, generate issues that you can then track and investigate.

#### **NOTE:**

Indicators should be present in the Threat Intelligence database (Threat Management → Threat Intelligence → Indicators) before creating detection rules.

You can create rules based on filters that are applied to a file (SHA256, MD5) an IP address, and a domain. If an indicator rule applies, an issue is generated in Cortex AgentiX (source is Threat Intelligence).

Indicator rules are designed to leverage threat intelligence indicators like MD5 and SHA256 hashes that are present in your TIM library. These rules directly integrate with and rely on the indicators ingested and managed by TIM. Indicators must be in the TIM database before creating these rules.

#### Create a Detection rule

After you create a detection rule, Cortex AgentiX searches for indicators in your tenant and raises an issue if a match is detected. Detection rules apply for File, Domain, and IP Address indicator types.

1. Select Threat Management → Detection Rules → Indicator Rules → Add Rule → Detection Rule.

2. From the Create New Detection Rule wizard, in the General section, add the following parameters:

Parameter	Description
Rule Name	Add a meaningful name.
Severity	Defines the severity of the issue.
Description	Add a meaningful description.

3. Click Next.

4. In the Target section, use the filters and/or select the file indicators to which to apply the rule.

#### **NOTE:**

You can't change the Detectable = True and Status = Active filters, which comply with the requirements of the supported indicator type for detection.

Filter	Description
Value	The hash value of the field (SHA256 or MD5), IP address, or domain.
Verdict	The reputation of the indicator: Malicious, Suspicious, Benign, Unknown
Has Created Issues	Whether the indicator has an issue that has been created.
Campaign	Whether the indicator is part of an existing campaign.
Mitre ID	Mitre ID associated with the related issues.



Filter	Description
Mitre Tactic	Mitre Tactic associated with the related issues.
Tags	The tags applied to indicators.
Confidence	The level of confidence.
Aggregated Reliability	The reliability score, such as A - Completely reliable.
Feed	The source (script, manual, etc.) that last set the indicator's expiration status.
Type	The indicator type (Domain, File, IP)

5. Click Next and then save the rule.

6. If the indicator rule has generated issues, right-click the rule and select View related issues.

Example 77. Create a detection rule from feeds

In this example, create a detection rule from many feeds, such as Unit 42, AzureRiskyUsers, and Mail-Sender, that returns a malicious verdict.

1. In the General section, add the following parameters.

Field	Value
Rule Name	JC-IR-Prevent-01
Severity	Medium
Description	To raise detection on all indicators uploaded from feeds with a malicious verdict.

2. In the Target Section, select Feed (Select All) and Verdict = Malicious.

TYPE	VALUE	VERDICT	FIRST SEEN	LAST SEEN	SOURCE TIME STAMP	RELATED ALERTS	
File	1a5ebc86749609fc1e9dd0c96db855b1bdb0995...	Malicious	Nov 17th 2023 14:0...	280...5861d@2942726	Nov 28th 2023 11:4...	2fb...95fda@2951494	1
Domain	http://xn--90aedbb0beifaojw.xn--p1ai/a/	Malicious	N/A	API	N/A	API	0
File	cde11875ac8312bb069dabc9b0502909755679...	Malicious	Nov 9th 2023 06:00...	d5f...6f876@2940808	Nov 25th 2023 02:1...	7f7...24023@2950894	1
File	fb077a7e7f7ba39af50419bcbc4669ed6b1ba32c...	Malicious	Nov 17th 2023 04:0...	ab3...98272@2942601	Nov 25th 2023 02:1...	10a...7dda0@2950893	2
File	d451d440e36c8197be5a8cb3896dee351b1962...	Malicious	Sep 8th 2023 17:26...	c36...b663a@2926966	Nov 22nd 2023 17:...	N/A	9
File	2CA2D550E603D74DEDDA03156023135B38D...	Malicious	Aug 2nd 2023 07:0...	291...6be0f@2910703	Nov 22nd 2023 17:...	N/A	1

When a malicious verdict is found from the feed, an issue is generated. The Issue Source is Threat Intelligence, severity is medium and the Action is Detected.



	ALERT SOURCE	CATEGORY	ALERT NAME	ALERT ID	TIMESTAMP	HOST	USER NAME	SEVERITY	ACTION
•	Threat Intelligence	Hash	JC-IR-Detect-01	2975797	Jan 3rd 2024 10:58:39	pmcbWin20161...	pmcbWin2016125\patr...	Medium	Detected
•	Threat Intelligence	Hash	JC-IR-Detect-01	2975794	Jan 3rd 2024 10:55:54	pmcbWin20161...	pmcbWin2016125\patr...	Medium	Detected
•	Threat Intelligence	Hash	JC-IR-Detect-01 - ea2bec8ec6a4f895679a203cb0b0a4d8	2975620	Jan 3rd 2024 04:00:46	pmcbWin20161...	patrick\pmcgreen	Medium	Detected
•	Threat Intelligence	Hash	JC-IR-Detect-01 - ed3973a8662c1716629099840eff9bf...	2975621	Jan 3rd 2024 04:00:46	pmcbWin20161...	patrick\pmcgreen	Medium	Detected

#### NOTE:

The Issue source is Threat Intelligence.

#### Manage Indicator Rules

The Indicator Rules page displays the following fields for each rule:

Field	Description
Rule ID	A unique identifier for the rule.
Creation Date	Timestamp of when the rule was created.
Modification Date	Timestamp when the rule was edited.
Name	Name of the rule.
Type	Whether the rule is a Prevention or Detection type rule.
Target	Hash, IP address, File, or domain value associated with the rule.
Severity	Level of severity associated with the rule.
# of issues	Number of issues generated by the rule.
Created by	The email address of the user who created the rule.
Description	An optional description associated with the rule.
Status	Whether the rule is Enabled or Disabled.

#### NOTE:

If an indicator matches multiple indicator rules, the highest severity rule is used. If all have the same severity, the rules are used by the first created.

In the Indicator Rules table, right-click a rule to perform actions, including the following:

Action	Description
View related issues	View issues generated by the rule.
Disable/Enable	Depending on the current status, Disable or Enable the rule.



Action	Description
Edit Rule	Modify the rule.
Save as new	Create a new rule using the current rule configurations.
Delete	Delete the rule.

### 5.2.8 | Export indicators

#### Abstract

Export indicators from the Indicators table, using an integration, or playbook, or set up an External Dynamic list (EDL) by using the Generic Export Indicators integration.

In the Indicators table, you can export indicators in a CSV or STIX file. You can also export indicators using an integration or a playbook.

#### Export indicators using the Generic Export Indicators Integration

You can export indicators in a hosted text file (External Dynamic list) from Cortex AgentiX or an engine using the Generic Export Indicators Service integration. Exported indicators can be used for example in firewall block lists, allow lists, and monitoring and analysis in Splunk. See Generic Export Indicators Service.

The Generic Export Indicators Service integration can be configured to export specific fields in different output formats. Multiple instances of the integration can be configured for different indicator queries, and the output can be customized to work with a variety of third-party services.

You can set up the Generic Export Indicators Service integration by setting up a long-running integration. See Forward Requests to Long-Running Integrations.

If you configure the Generic Export Indicator to run on-demand, use the `!export-indicators-list-update` command for the first time to initialize the export process.

#### Export indicators using playbooks

Cortex AgentiX provides out-of-the-box playbooks for TIM, including playbooks that enable you to export indicators. All TIM-related playbooks have the 'TIM' prefix. Some are generic and some are dedicated to a specific vendor, like QRadar (for example, TIM - QRadar Add Domain Indicators) and ArcSight (for example, TIM- Arcsight Add IP Indicators).

If you define a playbook task input that pulls from indicators, the entire playbook runs in Quiet Mode. This means the task or playbook information is not written to the War Room, and inputs and outputs are not displayed in the playbook. However, errors and warnings are still written to the War Room.

#### CAUTION:

You should not run a query on a field that you might change in the playbook flow. For example, you shouldn't have a playbook with query `Verdict:Malicious` and then change the indicator verdict as a part of the playbook.

### 5.3 | Indicator management

#### Abstract

Perform actions (create, edit, export, delete) and search for indicators on the Cortex AgentiX Indicators page.

Indicators are artifacts associated with security issues and are an essential part of the case management and remediation process. They help correlate issues, create hunting operations, and enable you to easily analyze cases and reduce Mean Time to Response (MTTR).

#### Indicators

Displays a list of indicators added to Cortex AgentiX, where you can perform several indicator actions.

You can perform the following actions on the AGENTIX Indicators page.

Action	Description
Investigate an indicator	Click on an indicator to view and take action on the indicator.



Action	Description
Create an indicator	Indicators are added to the indicators table from feed integration or you can manually create a new indicator in the system.  When creating an indicator, in the Verdict field, you can either select a verdict or leave it blank to calculate it later by clicking Save & Enrich, which updates the indicator from enrichment sources. After you select an indicator type, you can add any custom field data.
Edit	Edit a single indicator or select multiple indicators to perform a bulk edit.
Delete and Exclude	Delete and exclude one or more indicators from all indicator types or a subset of indicator types.  If you select the Do not add to exclusion list checkbox, the selected indicators are only deleted.
Export CSV	Export the selected indicators to a CSV file.
Export STIX	Export the selected indicators to a STIX file.
Upload a STIX file	To upload a STIX file, click the upload button (top right of the page) and add the indicators from the file to the system.

## 5.4 | Indicator investigation

### Abstract

Learn how to use TIM in your use case, investigating an indicator and creating indicator relationships.

Cortex AgentiX enables you to centralize and manage every aspect of your TIM investigation. Create, extract, and enrich indicators and explore their relationships to gain deeper insights.

After you start ingesting indicators into Cortex AgentiX, you can start your investigation, including creating indicators, adding indicators to an issue, extracting indicators, exporting indicators, etc.

When investigating an indicator, you can see the following tabs:

- Summary

View verdict, enrich, expire, delete and exclude the indicator, add relationships, view related issues, and add comments. Add or remove tags, which can help classify known threats. For example, you may want to group specific malware indicators that are part of ransomware, such as trojan or loader.

- Additional Details

Add or view any community notes for sharing and any custom details.

When investigating an indicator, you can perform actions on the indicator, such as:

Action	Description
Enrich an indicator	You can view detailed information about the indicator (WHOIS information for example), using third-party integrations such as VirusTotal and IPinfo. For more information, see Extract and enrich an indicator.
Expire an indicator	You may want to expire an indicator to filter out less relevant issues, allowing analysts to focus on active threats. For more information, see Expire an indicator.



Action	Description
Manage indicator relationships	Indicator relationships are connections between different indicators. These relationships can be IP addresses related to one another, domains impersonating legitimate domains, etc. Relationships are created from threat intel feeds and enrichment integrations that support the automatic creation of relationships. For more information, see Manage indicator relationships.
Delete and exclude indicators	Indicators added to an exclusion list are disregarded by the system and are not created or involved in automated flows. For more information, see Delete and exclude indicators.

#### 5.4.1 | Indicator verdict

##### Abstract

Cortex AgentX analyzes indicators to determine whether they are malicious. Create indicator types and custom layouts, exclusion lists, and indicator verdicts.

An indicator's verdict is assigned according to the verdict returned by the source with the highest reliability, where reliability is scaled based on the Admiralty Source and Information Reliability Matrix. In cases where multiple sources with the same reliability score return a different verdict for the indicator, the worst verdict is taken. Indicators are assigned the following verdicts:

- 0: Unknown
- 1: Benign
- 2: Suspicious
- 3: Malicious

In the UI, you can manually set the verdict when creating or editing an indicator. If you manually changed the indicator's verdict in the UI and want to recalculate it according to enrichment integrations, set the verdict to `Unknown` and then enrich the indicator. If you run indicator enrichment without setting the verdict to `Unknown`, the indicator is enriched but the manually set verdict is not changed.

You can also manually set the verdict by running `!setIndicator` or `!setIndicators` in the CLI (for example in the Playground), but if you set the verdict to `Unknown` the system will not overwrite it.

##### Source reliability

The reliability of an intelligence data source influences the verdict of an indicator and the values for indicator fields when merging indicators. Indicator fields are merged according to the source reliability hierarchy, which means that when there are two different values for a single indicator field, the field will be populated with the value provided by the source with the highest reliability score.

In rare cases, two sources with the same reliability score might return different values for the same indicator field. In these cases, the field is populated with the most recently provided source, unless the field is verdict. If two sources have the same reliability score and return different values for the verdict field, the worse verdict is used.

For the field types Tags and Multi-select, all values are appended, and nothing is overridden.

Source	Reliability Score	Notes
Manual	A+++	A user manually updates the verdict of an indicator.
Reputation script	A++	A script with the <code>reputation</code> tag calculates the verdict of an indicator. For example, the <code>DataDomainReputation</code> script evaluates the verdict of a URL or domain.
Third-party enrichment	A+	An integration or service that evaluates the verdict of an indicator. For example, the <code>urlscan.io</code> integration evaluates the verdict of a URL.



Source	Reliability Score	Notes
Feed	A: Completely reliable	The feed reliability is applied at the integration instance level.  For information on how to configure, see Configure Threat Intelligence feed integrations
	B: Usually reliable	
	C: Fairly reliable	
	D: Not usually reliable	
	E: Unreliable	
	F: Reliability cannot be judged	

#### Different verdicts from integrations

In this example, two third-party integrations, VirusTotal and AlienVault, return a different verdict for the same indicator. The indicator's verdict will be Malicious because VirusTotal's reliability score is higher than AlienVault.

Integration	Reliability	Verdict	Final Verdict
VirusTotal	C - Fairly reliable	Malicious	Malicious
AlienVault	D- Not usually reliable	Benign	

In this example, two sources with the same verdict score return a different verdict for the same indicator. The indicator's verdict will be Malicious because when two sources have the same reliability, the worse verdict applies.

Integration	Reliability	Verdict	Final Verdict
TAXII Feed	B - Usually reliable	Malicious	Malicious
CSV Feed	B - Usually reliable	Benign	

#### 5.4.2 | Extract and enrich an indicator

##### Abstract

How to extract and enrich an indicator in Cortex AgentiX.

Indicator extraction identifies indicators from different text sources in the system (such as War Room entries), extracts them, and creates indicators in Cortex AgentiX. After extraction, the indicators are enriched.

##### NOTE:

By default, system-wide automatic indicator extraction and enrichment is disabled. However, if you migrated from Cortex AgentiX 2.x to Cortex AgentiX 3.x, system-wide automatic indicator extraction and enrichment is enabled.

If you have a Threat Intel Management (TIM) Add-on, you can enable or disable automatic indicator extraction system-wide. Go to Settings → Configuration → General → Server Settings. In the Indicators section, enable Enable automatic indicator extraction and enrichment from issues.

Indicator enrichment takes the extracted indicator and provides detailed information about the indicator (WHOIS information for example), using third-party integrations such as VirusTotal and IPInfo.



If you want to extract an indicator manually, you can do the following:

- Run indicator extraction in the CLI by running one of the following commands:

Command	Description
extractIndicators	<p>If you want to extract indicators from non-War-Room-entry sources (such as extracting from files), use the <code>!extractIndicators</code> command from the CLI. Use the command to do the following:</p> <ul style="list-style-type: none"><li>Validate regex: Test a specific string to see if the relevant indicators are extracted correctly, such as a URL.</li><li>In a playbook or script. The command extracts indicators in a playbook or a script (non War Room source), and also creates and enriches them.</li></ul> <p>You can extract the following:</p> <ul style="list-style-type: none"><li>A specified entry (an entry ID)</li><li>Investigation (Investigation ID)</li><li>Text</li><li>File path</li></ul> <p>For example, type <code>!extractIndicators text="some text 1.1.1.1 something" auto-extract=inline</code>. The entry text contains the text of the indicators, which is extracted and enriched.</p> <p>You can also extract indicators by adding the auto-extract parameter with the script and the mode for which you are setting it up. For example: <code>!ReadFile entryId=826@101 auto-extract=inline</code>.</p> <p>Usually, when using the CLI, you want to disable indicator extraction. For example, if you return internal/private data to the War Room, and you do not want it to be extracted and enriched in third-party services, add <code>auto-extract=none</code> to your CLI command.</p>
enrichIndicators	<p>The <code>enrichIndicators</code> command is usually used when you want to batch enrich indicators. This command works on existing indicators only (it does not create them on its own). When running the command, the relevant enrichment command is triggered (such as <code>!ip</code>), which is based on the indicator type that is found. The data is saved to context and the indicator.</p> <p><b>NOTE:</b></p> <p>Triggering enrichment on a substantial number of indicators can take time (because it's activating all enrichment integrations per indicator) and can result in performance degradation.</p>
Reputation commands	<p>Reputation commands such as <code>!ip</code>, can be run for new indicators and indicators already in the system. If extraction is on, the data is saved both to the indicator and the issue's context. If not, then the data is saved only to the context because the mapping flow is always triggered in enrichment commands. The default configuration is set to none in playbook tasks for extraction.</p> <p><b>NOTE:</b></p> <p>Reputation commands, such as <code>!ip</code>, <code>!domain</code> can only be used when you configure and enable a reputation integration instance, such as VirusTotal and WHOIS.</p>

- Use the Enrich indicator button in the indicator layout. This is the same effect as running a reputation command.
- Run indicator enrichment in the Quick View window

If there is an enhancement script attached to the indicator type, in the indicator Quick View window, you can run a script to enrich an indicator. For example, the Domain indicator type uses the `DomainReputation` enhancement script. In an issue that contains a domain indicator type, click Quick View. In the Indicators tab, click Domain → Actions → DomainReputation.

You can also run the enhancement script in the CLI.

#### 5.4.3 | Expire an indicator

##### Abstract

Expire an indicator in the CLI or in the UI.



Indicators can have the Expiration Status field set to Active or Expired. When indicators expire, they still exist in Cortex AgentiX, meaning they are still displayed and you can still search for them. You may want to expire an indicator to filter out less relevant issues, allowing analysts to focus on active threats. Expiring IoCs that are no longer relevant helps ensure that security systems remain focused on current threats.

You can set up expiration in the indicator type, integration feed, or in a script. For more information, see Configure indicator expiration. When you manually expire an indicator, this overrides indicator extraction rules set in scripts, indicator types, and feeds.

You can expire indicators using the following methods:

- In the indicator layout by clicking Expire indicator.
- Use the `expireIndicators` command to change the expiration status to Expired for one or more indicators. This command accepts a comma-separated list of indicator values and supports multiple indicator types. For example, you can set the expiration status for an IP address, domain, and file hash: `!expireIndicators value=1.1.1.1,safeurl.com,45356A9DB614ED7161A3B9192E2F318D0AB5AD10`.
- Use the `!setIndicator` or for multiple indicators use the `!setIndicators` command to reset the indicators' expiration value. The value can also be set to `Never`, so that the indicators never expire. For example, `!setIndicators indicatorsValues=watson.com expiration=Never`.

#### **NOTE:**

You need to run these commands in the Case or Issue War Room.

### 5.4.4 | Manage indicator relationships

Abstract

How to use and create indicator relationships in Cortex AgentiX and how it benefits an investigation.

Indicator relationships are connections between different indicators. These relationships can be IP addresses related to one another, domains impersonating legitimate domains, etc. These relationships enable you to enhance investigations with information about indicators and how they might be connected to other issues or indicators. For example, if you have a phishing issue with several indicators, one of those indicators might lead to another indicator, which is a malicious threat actor. Once you know the threat actor, you can investigate to see the issues it was involved in, its known TTPs (tactics, techniques, and procedures), and other indicators that might be related to the threat actor. The initial issue which started as a phishing investigation immediately becomes a true positive and relates to a specific malicious entity.

Relationships are created from threat intel feeds and enrichment integrations that support the automatic creation of relationships, such as AlienVault OTX v2 and URLhaus, by selecting Create relationships in the integration settings. Based on the information that exists in the integrations, the relationships are formed.

You can view indicator relationships by clicking on the indicator from an issue, and then from the Quick View window click the Relationships tab.

#### Create indicator relationships

You can also manually create and modify relationships, which is useful when a specific threat report comes out. For example, Unit 42's SolarStorm report contains indicators and relationships that might not exist in your system, or you might not be aware of their connection.

If a relationship is no longer relevant, you can revoke it. This might be relevant, for example, if a known malicious domain is no longer associated with a specific IP address.

When you create a relationship, you can set the relationship type such as whether the indicator is related, attached, applied, etc. For example, a file is attached-to an email. The email communicated-with the file.

You can create relationships by adding them in a playbook, in the CLI using the `CreateIndicatorRelationship` command, or when investigating an indicator in the Threat Intel tab.

How to add an indicator relationship from an Indicator

1. Open an indicator and in the RELATIONSHIPS section add a relationship.
2. In the New Relationships window, in Step 1, add a query by which to search for the relevant indicators.  
You can optionally limit the time range for the search.
3. Select the indicators you want to create a relationship to.
4. In Step 2 set the relationship type.

By default, the relationship is related-to. For example, IP address x.x.x.x is related-to IP address y.y.y.y.

5. Save the relationship.

#### **NOTE:**

You can also add an indicator relationship from the Quick View when selecting an indicator from an issue.



## Investigate an indicator using indicator relationships

In this example, you can see how to use the relationships feature to further your investigation.

- When opening an issue, the severity is low, but the issue contains the following indicators:

- File
- IP

- When you click the file hash indicator, neither the Info nor Relationships tabs have any additional details. This seems to indicate that the file is harmless.

The screenshot shows the 'Relationships' tab for a file indicator with the ID 867670c365b5132b8c15ffec18ea609c. The tab bar includes 'Info', 'Relationships' (which is active), 'Unit 42 Intel', and 'Show empty fields'. Below the tab bar, there's a section for 'RELATIONSHIPS (0)' with a '+ Add' button and a 'Hide Revoked' checkbox. A message states 'Indicator does not have any relationships'.

- Click on the IP address indicator.

Under the Info tab, you can see that the indicator was ingested from a threat intel feed. This already bears further investigation.

The screenshot shows the 'Info' tab for an IP indicator. The indicator is labeled 'Active' and was 'Set by Indicator Type IP on May 25, 2021 at 1:12 PM'. Under the 'Reputation' section, it is listed as 'Unknown' (Set by @DBot). The 'Source' and 'Verdict' columns show 'Indicators feed - ...' and 'Unknown' respectively. The 'Source Time Stamp' is 'May 25, 2021, 1:12 PM'.

- Go to the Relationships tab.

You can see that this indicator is related to a campaign.

The screenshot shows the 'Relationships' tab for an IP indicator with the ID 212.114.52.148. The tab bar includes 'Info', 'Relationships' (which is active), and 'Show empty fields'. Below the tab bar, there's a section for 'Relationships (1)' with a '+ Add' button and a 'Hide Revoked' checkbox. A table lists one relationship: 'indicated-by' (Relationship) to 'Campaign 1 - Hangover BackConfig' (Related Indicator). The 'Indicator Type' is 'Campaign' and the 'Modified' date is 'May 25, 2021, 1:12 PM'.

What started as a low severity issue, has become a lot more threatening.

### 5.4.5 | Delete and exclude indicators

#### Abstract

Indicators added to an exclusion list are disregarded by the system. Add indicators to an exclusion list in Cortex AgentIX.

Indicators added to an exclusion list are disregarded by the system and are not created or involved in automated flows such as indicator extraction. You can still manually enrich IP addresses and URLs that are on the exclusion list, but the results are not posted to the War Room.

Add indicators to the exclusion list either in the Indicators table or in the Exclusion List page.

#### Delete and exclude indicators in the Indicators table

Select one or more indicators from the Indicators table and click the Delete and Exclude button. The indicators are deleted from the Indicators table and added to the exclusion list. You can associate these indicators with one or more indicator types.

If you delete the indicator it is removed from Cortex AgentIX. This option should be used mainly for correcting errors in ingestion, and not as part of your regular workflow.



## Add indicators in the Exclusion List page

From the Exclusion List page, you can view the list of excluded indicators, add an indicator to the exclusion list, or define indicator values to be excluded using a regular expression (regex) or CIDR.

1. Select Settings → Configurations → Object Setup → Indicators → Exclusion List → New excluded indicator.

2. Add the indicator value. For example, example.com (for a domain).

### CAUTION:

Ensure you are using the correct syntax when defining the values for your exclusion lists.

3. Select whether to use Regex.

A regular expression enables you to identify a sequence of characters in an unknown string. The following example would identify www.demisto.com:  
[A-Za-z0-9!@#\$%^&]\*demisto[A-Za-z0-9!@#\$%^&]\*.

Classless inter-domain routing (CIDR) enables you to define a range of IP addresses. For example, the IPv4 block 192.168.100.0/22 represents the 1024 IPv4 addresses from 192.168.100.0 to 192.168.103.255.

4. Add a reason as to why you are excluding the indicator.

5. Add the indicator types that apply.

6. Save the excluded indicator.

## Exclusion list examples

Exclusion	Description	Settings
Domain, URLs, and subdomains	Excludes a specific domain, and all subdomains and URLs associated with the domain.	<p>Define two entries to cover all URLs and subdomains associated with a specific domain.</p> <p>Entry one:</p> <ul style="list-style-type: none"><li>Value: Subdomains and URLs. Example: \.example\.com</li><li>Select Use Regex.</li><li>Do not select any indicator types.</li></ul> <p>Entry two:</p> <ul style="list-style-type: none"><li>Value: The specific domain. Example: example.com</li><li>Do NOT select Use Regex.</li><li>Do not select any indicator types.</li></ul>
Subdomain (and URLs) specifically	Excludes any subdomains and URLs of a domain, but the domain is still extracted.	<ul style="list-style-type: none"><li>Value: Subdomains and URLs. Example: \.example\.com</li><li>Select Use Regex.</li><li>Do not select any indicator types.</li></ul>
Specific domain only	Excludes a specific domain. Subdomains and URLs are still extracted.	<ul style="list-style-type: none"><li>Value: The specific domain. Example: example.com</li><li>Do NOT select Use Regex.</li><li>Select indicator type: Domain.</li></ul>
URL with wildcards	Excludes any indicators of type URL matching the regex. Indicators example.com and examplesub.example.com of type Domain would still be extracted. Start the regex with https?:// to exclude both HTTP and HTTPS URLs.	<ul style="list-style-type: none"><li>Value: The URL with wildcard added at the end. Example: http://examplesub.example.com</li><li>Select Use Regex.</li><li>Select indicator type: URL.</li></ul>



Exclusion	Description	Settings
Specific URL	Excludes a specific URL, but the domain and subdomains are still extracted.	<ul style="list-style-type: none"> <li>Value: The specific URL. Example: <code>http://examplesub.example.com/myexample</code></li> <li>Do NOT select Use Regex.</li> <li>Select indicator type: URL.</li> </ul>
URLs, domain, and subdomains, case-insensitive, anchored to start	Excludes domain example.com, its subdomains, and its URLs. Case-insensitive. Anchors regex match to the start of the indicator value, so indicators that contain but do not start with a match (e.g., example.net? param@example.com) are not excluded.	<ul style="list-style-type: none"> <li>Value: Domain, subdomains and URLs, case insensitive and anchored to the start of the indicator. Example: (?i)^(<a href="https://">//)?(([a-zA-Z0-9\-.]+\.)+)?example\.com</a></li> <li>Select Use Regex.</li> <li>Select indicator types: URL, Domain.</li> </ul>
All URLs	Excludes all URLs for a specific domain that have a path (even an empty path), but the domain and subdomains are still extracted.	<ul style="list-style-type: none"> <li>Value: URLs with or without a path. Example: <code>example\.com/</code></li> <li>Select Use Regex.</li> <li>Do not select any indicator types.</li> </ul>

## 6 | Cortex AgentiX XQL

Understand more about the Cortex Query Language called XQL, so you can build queries to gain insight from the data contained in the different data sources in Cortex AgentiX.

### 6.1 | Get started with XQL

#### Abstract

XQL is the Palo Alto Networks Cortex Query Language used in Cortex AgentiX.

XQL is the Cortex Query Language. It allows you to form complex queries against data stored in Cortex AgentiX. This section introduces XQL, and it provides reference information on the various stages, functions, and aggregates that XQL supports.

#### 6.1.1 | XQL language features

#### Abstract

Learn more about the Cortex Query Language features to query for raw network and endpoint data.

The Cortex Query Language (XQL) enables you to query for information contained in a wide variety of data sources in Cortex AgentiX for rigorous endpoint and network event analysis. Queries require a dataset, or data source, to run against. In a dataset query, unless otherwise specified, the query runs against the `xdr_data` dataset, which contains all raw log information that Cortex AgentiX collects from all Cortex product agents, including EDR data, and PAN NGFW data. In XDM queries, you must specify the dataset mapped to the XDM that you want to run your query against. For both types of queries, you can also import data from third parties and then query against those datasets as well.

You submit XQL queries to Cortex AgentiX using the Investigation & Response â Search â Query Builder user interface.

XQL is similar to other query languages, and it uses some of the same functions as can be found in many SQL implementations, but it is not SQL. XQL forms queries in stages. Each stage performs a specific query operation and is separated by a pipe (|) character. To help you create an effective XQL query with the proper syntax, the query field in the user interface provides suggestions and definitions as you type. For example, the following dataset query uses three stages to identify the dataset to query, identify the field to be retrieved from the dataset, and then set a filter that identifies which records should be retrieved as part of the query:

```
dataset = xdr_data
| fields os_actor_process_file_size as osapfs
| filter to_string(osapfs) = "12345"
```

#### TIP:

When creating XQL queries, you can:



- Use the up and down arrow keys to navigate through the auto-suggestion commands and definitions.
- Select an auto-suggestion command by pressing either the Enter or Tab key.
- Press Shift+Enter to add a new line, and easily ignore the auto-suggestion output.
- Close the auto-suggestion output by pressing the Esc key.

XQL supports:

- Simple queries.
- Filters that identify a subset of records to return in the result set.
- Joins and Unions.
- Aggregations.
- Queries against standard datasets.
- Queries against presets, which are collections of information that are specific to a given type of network or endpoint activity, such as authentication or file transfers.
- Queries against custom imported datasets.
- Queries against the XDM.

## 6.1.2 | XQL Language Structure

Abstract

Learn more about the Cortex Query Language structure when creating a query.

Cortex Query Language (XQL) queries usually begin by defining a data source, be it a dataset, preset, or Cortex Data Model (XDM). You must specify the dataset mapped to the XDM that you want to run your query against. In a dataset query, unless otherwise specified, the query runs against the `xdr_data` dataset, which contains all log information that Cortex AgentX collects from all Cortex product agents, including EDR data, and PAN NGFW data. It's possible to change the default dataset in the Dataset Management page of Cortex AgentX. For more information, see [What are datasets?](#).

After specifying a data source, you use zero or more stages to form the XQL query. Each stage is delimited using a pipe character (|). The function performed by each stage is identified by the stage keyword that you provide. XQL queries can contain different components depending on the type of query you want to build.

### 6.1.2.1 | Adding comments in queries

Abstract

Learn more about adding comments in Cortex Query Language queries.

You can add comments in any section when building a query in Cortex Query Language (XQL).

- Comments are added on a single line using the following syntax.

```
//<comments>
```

For example,

```
dataset = xdr_data
| filter event_type=1
//ENUM.process
and event_sub_type = 1
//ENUM.execution
```

- To write a comment that extends over multiple lines use the following syntax.

```
/*multi-line <comments> */
```

For example,

```
dataset = xdr_data
| filter
/*multi-line Adding comments is a great thing.
Here is an example */
event_type=1
```

## 6.1.3 | Supported operators

Abstract



Cortex Query Language supports specific comparison, boolean, and set operators in Cortex AgentIX.

Cortex Query Language (XQL) queries support the following comparison, boolean, string, range, and add operators.

Operator	Description
Comparison operators	
=, !=	Equal, Not equal
<, <=	Less than, Less than or equal to
>, >=	Greater than, Greater than or equal to
Boolean operators	
and	Boolean and
or	Boolean or
not	Boolean not
String and range operators	
IN, NOT IN	<p>Returns true if the integer or string field value is one of the options specified. For example:</p> <pre>action_local_port in(5900,5999)</pre> <p>For string field values, wildcards are supported. In this example a wildcard (*) is used to search if the value contains the strings "word_1" or "word_2" anywhere in the output, or exactly matches the string "word":</p> <pre>str_field in ("*word_1*", "*word_2*", "word")</pre> <p><b>NOTE:</b></p> <p>In some cases, using an IN or NOT IN operator combined with a dataset and filter stage can be a better alternative to using a join stage.</p>
CONTAINS, NOT CONTAINS	<p>Performs a search for an integer or string. Returns true if the specified string is contained in the field. Contains and Not Contains are also supported within arrays for integers and strings.</p> <p>Example 78.</p> <pre>lowercase(actor_process_image_name) contains "psexec"</pre>
~=	<p>Matches a regular expression.</p> <p>Example 79.</p> <pre>action_process_image_name ~= ".*\?.\.(?:pdf docx)\.exe"</pre>



Operator	Description
INCIDR, NOT INCIDR	<p>Performs a search for an IPv4 address or IPv4 range using CIDR notation, and returns true if the address is in range.</p> <p>Example 80.</p> <pre data-bbox="271 316 601 339">action_remote_ip incidr "192.1.1.1/24"</pre> <p>It is also possible to define multiple CIDRs with comma separated syntax when building a XQL query with the Query Builder or in Correlation Rules. When defining multiple CIDRs, the logical OR is used between the CIDRS listed, so as long as one address is in range the entire statement returns <code>true</code>. The same logic is used when using the <code>incidr()</code> function. For more information on how this logic works to determine whether the <code>incidr</code> or <code>not incidr</code> operators return <code>true</code> or <code>false</code>, see <code>incidr</code>.</p> <p>Example 81.</p> <pre data-bbox="271 586 744 609">action_remote_ip incidr "192.168.0.0/24, 1.168.0.0/24"</pre> <p>Both the IPv4 address and CIDR ranges can be either an explicit string using quotes (" "), such as "192.168.0.1", or a string field.</p>
INCIDR6, NOT INCIDR6	<p>Performs a search for an IPv6 address or IPv6 range using CIDR notation, and returns true if the address is in range.</p> <p>Example 82.</p> <pre data-bbox="271 856 903 878">action_remote_ip incidr6 "3031:3233:3435:3637:0000:0000:0000:0000/64"</pre> <p>It is also possible to define multiple CIDRs with comma separated syntax when building a XQL query with the Query Builder or in Correlation Rules. When defining multiple CIDRs, the logical OR is used between the CIDRS listed, so as long as one address is in range the entire statement returns <code>true</code>. The same logic is used when using the <code>incidr6()</code> function. For more information on how this logic works to determine whether the <code>incidr6</code> or <code>not incidr6</code> operators return <code>true</code> or <code>false</code>, see <code>incidr6</code>.</p> <p>Example 83.</p> <pre data-bbox="271 1126 960 1148">action_remote_ip incidr6 "2001:0db8:85a3:0000:0000:8a2e:0000:0000/64, fe80::/10"</pre> <p>Both the IPv6 address and CIDR ranges can be either an explicit string using quotes (" "), such as "3031:3233:3435:3637:0000:0000:0000:0000/64", or a string field.</p>
Add operator for tagging	
add	<p>The <code>add</code> operator is used in combination with the <code>tag</code> command to add a single tag or list of tags to a field that you can easily query in the dataset.</p> <p>Example 84.</p> <ul data-bbox="303 1553 505 1575" style="list-style-type: none"> <li>• Adding a Single Tag</li> </ul> <pre data-bbox="319 1598 489 1643">dataset = xdr_data   tag add "test"</pre> <ul data-bbox="303 1665 515 1688" style="list-style-type: none"> <li>• Adding a List of Tags</li> </ul> <pre data-bbox="319 1710 632 1755">dataset = xdr_data   tag add "test1", "test2", "test3"</pre>

#### 6.1.4 | Datasets and presets

##### Abstract

The Cortex Query Language supports built-in datasets, custom datasets, and presets.

Every Cortex Query Language (XQL) dataset query begins by identifying a data source that the query will run against. Each data source has a unique name, and a series of fields. Your query specifies the data source, and then provides stages that identify fields of interest and perform operations against those fields.



You can query against either datasets or Presets in a dataset query. XQL supports using different languages for dataset and field names. In addition, the dataset formats supported are dependent on the data retention offerings available in Cortex AgentiX according to whether you want to query hot storage (default) or cold storage. For more information, see [XQL Language Structure](#).

#### Datasets

The standard, built-in data source that is available in every Cortex AgentiX instance is the `xdr_data` dataset. This is a very large dataset with many available fields. For more information about this dataset, see [Cortex XQL Schema Reference](#). Cortex Query Language (XQL) supports using different languages for dataset and field names. In addition, the dataset formats supported are dependent on the data retention offerings available in Cortex AgentiX according to whether you want to query hot storage (default) or cold storage. For more information, see [XQL Language Structure](#).

This dataset is comprised of both raw Endpoint Detection and Response (EDR) events reported by the Cortex AgentiX agent, and of logs from different sources such as third-party logs. To help you investigate events more efficiently, Cortex AgentiX also stitches these logs and events together into common schemas called stories. These stories are available using the Cortex AgentiX Presets.

#### Building queries in XQL

When building queries in XQL, keep the following in mind about datasets:

- Use the `dataset` keyword to specify a dataset on your query.
- Create custom datasets using the target stage.
- Dataset names can use uppercase characters, but in queries dataset names are always treated as if they are lowercase. In addition, dataset names are supported using different languages, numbers (0–9), and underscores (\_). Yet, underscores cannot be the first character of the name.
- Upon ingestion, all fields are retained even fields with a null value. You can also use XQL to query parsing rules for null values.
- Schema changes to datasets may not be reflected in the autocomplete suggestions and definitions as you type in real time the XQL query and can appear with a slight delay.

#### Available datasets

Depending on your integrations, you can have the following datasets available for queries:

Data	Dataset
Issues table in Cortex AgentiX	<p>issues</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"><li>• INFO issues are not included in this dataset.</li><li>• This dataset includes issues from the Security and Health domains. For more information, see <a href="#">Overview of the Issues page</a>.</li></ul>
Authentication logs (subset of <code>xdr_data</code> )	<p>Authentication logs, such as Okta: <code>auth_logs</code></p> <p><b>NOTE:</b></p> <p>The fields contained in this dataset are a subset of the fields in the <code>xdr_data</code> dataset.</p>
AWS CloudTrail and Amazon CloudWatch	<Vendor>_<Product>_raw
Azure Event Hub	<ul style="list-style-type: none"><li>• All logs: <code>MSFT_Azure_raw</code></li><li>• Normalize and enrich audit logs: <code>cloud_audit_logs</code></li></ul>
Azure Network Watcher	<ul style="list-style-type: none"><li>• All logs: <code>MSFT_Azure_raw</code></li><li>• Normalize and enrich flow logs: <code>xdr_dataset</code> dataset with a preset called <code>network_story</code></li></ul>
BeyondTrust Privilege Management Cloud	<code>beyondtrust_privilege_management_raw</code>



Data	Dataset
Box	<p>Events (admin_logs)</p> <ul style="list-style-type: none"> <li>• <code>box_admin_logs_raw</code></li> </ul> <p>Box Shield Alerts</p> <ul style="list-style-type: none"> <li>• <code>box_shield_alerts_raw</code></li> </ul> <p>Users</p> <ul style="list-style-type: none"> <li>• <code>box_users_raw</code></li> </ul> <p>Groups</p> <ul style="list-style-type: none"> <li>• <code>box_groups_raw</code></li> </ul>
Checkpoint FW1/VPN1	<code>&lt;Vendor&gt;_&lt;Product&gt;_raw</code>
Cisco ASA	<p>Cisco ASA firewalls or Cisco AnyConnect VPN</p> <ul style="list-style-type: none"> <li>• <code>cisco_asa_raw</code></li> </ul>
Collector status change audit for collection integrations, custom collectors, and marketplace collectors.	<code>collection_auditing</code>
Corelight Zeek	<code>corelight_zeek_raw</code>
Correlation rule executions	<code>correlations_auditing</code>
Cortex Data Lakes	<code>xdr_data</code>
Cortex XDR Collectors	<code>panw_xdrc_raw</code>
Cortex AgentX Host Firewall enforcement events	<code>host_firewall_events</code>
CrowdStrike FDR	<ul style="list-style-type: none"> <li>• <code>crowdstrike_falcon_incident_raw</code></li> <li>• <code>crowdstrike_fdr_raw</code></li> </ul>
CSV files in shared Windows directory	Custom datasets: Select from pre-existing user-created datasets or add a new dataset.
Database data (MySQL, PostgreSQL, MSSQL, and Oracle)	<code>&lt;Vendor&gt;_&lt;Product&gt;_raw</code>



Data	Dataset
Data ingestion health metrics	<p>Datasets:</p> <ul style="list-style-type: none"> <li>• <code>data_ingestion_health</code></li> </ul> <p><b>IMPORTANT:</b></p> <p>This dataset will not be updated after June 2024. Use the <code>health_alerts</code> dataset instead.</p> <ul style="list-style-type: none"> <li>• <code>metrics_source</code></li> </ul> <p>Presets:</p> <ul style="list-style-type: none"> <li>• <code>data_ingestion_metrics</code> (this preset will be deprecated in the next release and replaced by <code>metrics_view</code>).</li> <li>• <code>metrics_view</code></li> </ul>
Dropbox	<p>Events</p> <ul style="list-style-type: none"> <li>• <code>dropbox_events_raw</code></li> </ul> <p>Member Devices</p> <ul style="list-style-type: none"> <li>• <code>dropbox_members_devices_raw</code></li> </ul> <p>Users</p> <ul style="list-style-type: none"> <li>• <code>dropbox_users_raw</code></li> </ul> <p>Groups</p> <ul style="list-style-type: none"> <li>• <code>dropbox_groups_raw</code></li> </ul>
Elasticsearch Filebeat	<code>&lt;Vendor&gt;_&lt;Product&gt;_raw</code>
Elasticsearch Winlogbeat	<code>&lt;Vendor&gt;_&lt;Product&gt;_raw</code> <p><b>NOTE:</b></p> <p>If the vendor and product are not specified in the Winlogbeat profile's configuration file, Cortex AgentiX creates a default dataset called <code>microsoft_windows_raw</code>.</p>
Errors related to Parsing Rules and Data Model Rules	<code>parsing_rules_errors</code>
Errors related to event forwarding	<code>event_forwarding_errors</code>
Forcepoint DLP	<code>forcepoint_dlp_endpoint_raw</code>
Fortinet Fortigate	<code>&lt;Vendor&gt;_&lt;Product&gt;_raw</code>
GlobalProtect access authentication logs	<code>xdr_data</code> <p><b>NOTE:</b></p> <p>To ensure GlobalProtect access authentication logs are sent to Cortex AgentiX, verify that your PANW firewall's Log Settings for GlobalProtect has the Cortex Data Lake checkbox selected.</p>



Data	Dataset
Google Cloud Platform (GCP) logs	<ul style="list-style-type: none"> <li>All log types: <code>google_cloud_logging_raw</code></li> <li>Normalize and enrich audit and flow logs: <code>cloud_audit_logs</code> <ul style="list-style-type: none"> <li>Audit logs: <code>cloud_audit_logs</code></li> <li>Network flow logs: <code>xdr_dataset</code> dataset with a preset called <code>network_story</code></li> </ul> </li> </ul>
Google Kubernetes Engine (GKE)	<code>&lt;Vendor&gt;_&lt;Product&gt;_raw</code>
Google Workspace	<ul style="list-style-type: none"> <li>Google Chrome: <code>google_workspace_chrome_raw</code></li> <li>Admin Console: <code>google_workspace_admin_console_raw</code></li> <li>Google Chat: <code>google_workspace_chat_raw</code></li> <li>Enterprise Groups: <code>google_workspace_enterprise_groups_raw</code></li> <li>Login: <code>google_workspace_login_raw</code></li> <li>Rules: <code>google_workspace_rules_raw</code></li> <li>Google drive: <code>google_workspace_drive_raw</code></li> <li>Token: <code>google_workspace_token_raw</code></li> <li>User Accounts: <code>google_workspace_user_accounts_raw</code></li> <li>SAML: <code>google_workspace_saml_raw</code></li> <li>Alerts: <code>google_workspace_alerts_raw</code></li> <li>Emails: <code>google_gmail_raw</code></li> </ul>



Data	Dataset
Host Inventory and Vulnerability Assessment	<ul style="list-style-type: none"> <li>• Datasets <ul style="list-style-type: none"> <li>◦ <code>host_inventory</code></li> <li>◦ <code>va_cves</code></li> <li>◦ <code>va_endpoints</code></li> </ul> </li> <li>• Presets <ul style="list-style-type: none"> <li>◦ <code>host_inventory</code></li> <li>◦ <code>host_inventory_accessibility</code></li> <li>◦ <code>host_inventory_applications</code></li> <li>◦ <code>host_inventory_auto_runs</code></li> <li>◦ <code>host_inventory_cpus</code></li> <li>◦ <code>host_inventory_daemons</code></li> <li>◦ <code>host_inventory_disks</code></li> <li>◦ <code>host_inventory_drivers</code></li> <li>◦ <code>host_inventory_endpoints</code></li> <li>◦ <code>host_inventory_extensions</code></li> <li>◦ <code>host_inventory_groups</code></li> <li>◦ <code>host_inventory_kbs</code></li> <li>◦ <code>host_inventory_mounts</code></li> <li>◦ <code>host_inventory_services</code></li> <li>◦ <code>host_inventory_shares</code></li> <li>◦ <code>host_inventory_users</code></li> <li>◦ <code>host_inventory_volumes</code></li> <li>◦ <code>host_inventory_vss</code></li> </ul> </li> </ul>
Cases table in Cortex AgentiX	<code>cases</code>
Indicators	<code>indicators</code>
IT performance metrics	<code>it_metrics</code>
JSON or text logs from third-party source over HTTP	<code>&lt;Vendor&gt;_&lt;Product&gt;_raw</code>
Login logs (subset of <code>xdr_data</code> )	<p>Login logs, such as WEC: <code>login_logs</code></p> <p><b>NOTE:</b></p> <p>The fields contained in this dataset are a subset of the fields in the <code>xdr_data</code> dataset.</p>
Logs from third party source over FTP, FTPS, or SFTP	<code>&lt;Vendor&gt;_&lt;Product&gt;_raw</code>



Data	Dataset
Microsoft Defender for Endpoint	<code>msft_defender_raw</code>
Microsoft 365 (email)	<ul style="list-style-type: none"> <li>• <code>msft_o365_emails_raw</code></li> <li>• <code>msft_o365_users_raw</code></li> <li>• <code>msft_o365_groups_raw</code></li> <li>• <code>msft_o365_devices_raw</code></li> <li>• <code>msft_o365_mailboxes_raw</code></li> <li>• <code>msft_o365_rules_raw</code></li> <li>• <code>msft_o365_contacts_raw</code></li> </ul>
Microsoft Office 365	<ul style="list-style-type: none"> <li>• Microsoft Office 365 audit events from Management Activity API: <ul style="list-style-type: none"> <li>◦ Azure AD Activity Logs: <code>msft_o365_azure_ad_raw</code></li> <li>◦ Exchange Online: <code>msft_o365_exchange_online_raw</code></li> <li>◦ Sharepoint Online: <code>msft_o365_sharepoint_online_raw</code></li> <li>◦ DLP: <code>msft_o365_dlp_raw</code></li> <li>◦ General: <code>msft_o365_general_raw</code></li> </ul> </li> <li>• Microsoft Office 365 emails via Microsoft's Graph API: <code>msft_o365_emails_raw</code></li> <li>• Azure AD authentication events from Microsoft Graph API: <code>msft_azure_ad_raw</code></li> <li>• Azure AD audit events from Microsoft Graph API: <code>msft_azure_ad_audit_raw</code></li> <li>• Alerts from Microsoft Graph Security API: <code>msft_graph_security_alerts_raw</code></li> </ul>
NetFlow	<ul style="list-style-type: none"> <li>• <code>ip_flow_ip_flow_raw</code> (default)</li> <li>• When configured, uses the format &lt;Vendor&gt;_&lt;Product&gt;_raw</li> </ul>
Network Share logs	<Vendor>_<Product>_raw
Okta	<code>okta_sso_raw</code>
OneLogin	<p>Log collection</p> <ul style="list-style-type: none"> <li>• <code>onelogin_events_raw</code></li> </ul> <p>Directory</p> <ul style="list-style-type: none"> <li>• <code>onelogin_users_raw</code></li> <li>• <code>onelogin_groups_raw</code></li> <li>• <code>onelogin_apps_raw</code></li> </ul>
PANW EDR	<code>xdr_data</code>



Data	Dataset
PANW IOT Security	<p>Alerts</p> <ul style="list-style-type: none"> <li>• <code>panw_iot_security_alerts_raw</code></li> </ul> <p>Devices</p> <ul style="list-style-type: none"> <li>• <code>panw_iot_security_devices_raw</code></li> </ul>
PANW NGFW	<p><code>panw_ngfw_*_raw</code></p> <p>Supports the following logs.</p> <ul style="list-style-type: none"> <li>• Authentication Logs: <code>panw_ngfw_auth_raw</code></li> <li>• Configuration Logs: <code>panw_ngfw_config_raw</code></li> </ul> <p><b>NOTE:</b></p> <p>Prisma Access firewalls do not send configuration logs to the Structured Log Storage (SLS).</p> <ul style="list-style-type: none"> <li>• File Data Logs: <code>panw_ngfw_filedata_raw</code></li> <li>• Global Protect Logs: <code>panw_ngfw_globalprotect_raw</code></li> <li>• *Hipmatch Logs: <code>panw_ngfw_hipmatch_raw</code></li> <li>• System Logs: <code>panw_ngfw_system_raw</code></li> <li>• *Threat Logs: <code>panw_ngfw_threat_raw</code></li> <li>• *Traffic Logs: <code>panw_ngfw_traffic_raw</code></li> <li>• *URL Logs: <code>panw_ngfw_url_raw</code></li> <li>• User ID Logs: <code>panw_ngfw_userid_raw</code></li> <li>• Tunnel Logs: <code>panw_ngfw_tunnel_raw</code></li> <li>• Configuration Logs: <code>panw_ngfw_config_raw</code></li> </ul> <p>*These datasets use the query field names as described in the Cortex schema documentation.</p>
PingFederate	<code>ping_identity_pingfederate_raw</code>
PingOne for Enterprise	<code>pingone_sso_raw</code>
Playbook runs	<code>playbook_runs</code>
Playbook tasks	<code>playbook_tasks</code>
Prisma Browser	<code>panw_prisma_access_browser_raw</code>
Prisma Cloud	<code>prisma_cloud_raw</code>
Prisma Cloud Compute	<code>prisma_cloud_compute_raw</code>
Proofpoint Targeted Attack Protection	<code>proofpoint_tap_raw</code>



Data	Dataset
Scripts and commands metrics	<code>scripts_and_commands_metrics</code>
SentinelOne DeepVisibility	<code>sentinelone_deep_visibility_raw</code>
ServiceNow CMDB	A ServiceNow CMDB dataset is created for each table configured for data collection using the format <code>servicenow_cmdb_&lt;table name&gt;_raw</code> .
Salesforce.com	<ul style="list-style-type: none"> <li>• <code>salesforce_connectedapplication_raw</code></li> <li>• <code>salesforce_permissionset_raw</code></li> <li>• <code>salesforce_profile_raw</code></li> <li>• <code>salesforce_groupmember_raw</code></li> <li>• <code>salesforce_group_raw</code></li> <li>• <code>salesforce_user_raw</code></li> <li>• <code>salesforce_userrole_raw</code></li> <li>• <code>salesforce_document_raw</code></li> <li>• <code>salesforce_contentfolder_raw</code></li> <li>• <code>salesforce_attachment_raw</code></li> <li>• <code>salesforce_contentdistribution_raw</code></li> <li>• <code>salesforce_tenantsecuritylogin_raw</code></li> <li>• <code>salesforce_useraccountteammember_raw</code></li> <li>• <code>salesforce_tenantsecurityuserperm_raw</code></li> <li>• <code>salesforce_account_raw</code></li> <li>• <code>salesforce_audit_raw</code></li> <li>• <code>salesforce_login_raw</code></li> <li>• <code>salesforce_eventlogfile_raw</code></li> </ul>
Syslog/CEF	<code>&lt;CEFVendor&gt;_&lt;CEFP product&gt;_raw</code>
USB devices connect and disconnect events reported by the agent	<p><code>xdr_data</code></p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>• You can query in XQL for this data and build widgets based on the <code>xdr_data</code> dataset or using the preset <code>device_control</code>.</li> <li>• To view in an XQL query these events, the Device Configuration of the endpoint profile must be set to Block. Otherwise, the USB events are not captured. The events are also captured when a group of device types are blocked on the endpoints with a permanent or temporary exception in place. For more information, see [Ingest Connect and Disconnect Events of USB Devices] in Device control.</li> </ul>
VPN logs (subset of <code>xdr_data</code> )	<p>VPN logs, such as GlobalProtect: <code>vpn_logs</code></p> <p><b>NOTE:</b></p> <p>The fields contained in this dataset are a subset of the fields in the <code>xdr_data</code> dataset.</p>



Data	Dataset
Windows Endpoints using Cortex XDR Forensics Add-on	<ul style="list-style-type: none"> <li>• forensics_amcache</li> <li>• forensics_application_resource_usage</li> <li>• forensics_arp_cache</li> <li>• forensics_background_activity_monitor</li> <li>• forensics_chrome_history</li> <li>• forensics_cid_size_mru</li> <li>• forensics_command_history</li> <li>• forensics_dns_cache</li> <li>• forensics_edge_anaheim_history</li> <li>• forensics_edge_spartan_history</li> <li>• forensics_event_log</li> <li>• forensics_file_access</li> <li>• forensics_file_listing</li> <li>• forensics_firefox_history</li> <li>• forensics_handles</li> <li>• forensics_hosts_file</li> <li>• forensics_internet_explorer_history</li> <li>• forensics_jumplist</li> <li>• forensics_last_visited_pidl_mru</li> <li>• forensics_log_me_in</li> <li>• forensics_net_sessions</li> <li>• forensics_network</li> <li>• forensics_network_connectivity_usage</li> <li>• forensics_network_data_usage</li> <li>• forensics_open_save_pidl_mru</li> <li>• forensics_port_listing</li> <li>• forensics_prefetch</li> <li>• forensics_process_execution</li> <li>• forensics_process_listing</li> <li>• forensics_psreadline</li> <li>• forensics_recent_files</li> <li>• forensics_recentfilecache</li> <li>• forensics_recycle_bin</li> <li>• forensics_registry</li> <li>• forensics_remote_access</li> <li>• forensics_seven_zip_folder_history</li> <li>• forensics_shellbags</li> <li>• forensics_shimcache</li> </ul>



Data	Dataset
	<ul style="list-style-type: none"> <li>• <code>forensics_team_viewer</code></li> <li>• <code>forensics_typed_paths</code></li> <li>• <code>forensics_typed_urls</code></li> <li>• <code>forensics_user_access_logging</code></li> <li>• <code>forensics_user_assist</code></li> <li>• <code>forensics_windows_activities</code></li> <li>• <code>forensics_winrar_arc_history</code></li> <li>• <code>forensics_word_wheel_query</code></li> </ul>
Windows event logs via Cortex XDR Windows agents	<code>microsoft_windows_raw</code>
Windows Event Collector (WEC)	<ul style="list-style-type: none"> <li>• <code>xdr_data</code></li> <li>• <code>microsoft_windows_raw</code></li> </ul>
Windows DHCP using Elasticsearch Filebeat	<code>microsoft_dhcp_raw</code>
Windows DNS Debug using Elasticsearch Filebeat	<p>Raw Data</p> <ul style="list-style-type: none"> <li>• <code>microsoft_dns_raw</code></li> </ul> <p>Normalized Stories</p> <ul style="list-style-type: none"> <li>• <code>xdr_data</code> with the preset called <code>network_story</code>.</li> </ul>
Workday	<code>workday_workday_raw</code>
Zscaler Cloud Firewall	<p>ZIA</p> <ul style="list-style-type: none"> <li>• Firewall logs: <code>zscaler_nssfwlog_raw</code></li> <li>• Web logs: <code>zscaler_nssweblog_raw</code></li> </ul> <p>ZPA</p> <ul style="list-style-type: none"> <li>• <code>zscaler_zpa_raw</code></li> </ul>

#### Presets

Presets offer groupings of `xdr_data` fields that are useful for analyzing specific areas of network and endpoint activity. All of the fields available for a preset are also available on the larger `xdr_data` dataset, but by using the preset your query can run more efficiently. Presets are sorted at random by the first one million results found.

Two of the available presets are stories. These contain information stitched together from Cortex AgentX agent events and log files to form a common schema. They are `authentication_story` and `network_story`.

You use the `preset` keyword to specify a dataset in your query.

#### 6.1.5 | About examples

##### Abstract

Learn more about the Cortex Query Language (XQL) examples provided.



The examples included in the topics are intended to illustrate the behavior or usage of a particular stage or function. While these examples can be based on real data that you could use on real-world queries, you may need to tweak these queries to perform investigations or otherwise solve real-world problems.

For examples of queries that illustrate useful investigative queries, see the example Query Library that is available from the product user interface:

Investigation & Response ▾ Search ▾ Query Builder ▾ XQL ▾ Query Library

### 6.1.6 | JSON functions

#### Abstract

Learn more about how Cortex AgentIX treats JSON functions in the Cortex Query Language.

The Cortex Query Language (XQL) includes a number of JSON functions. Before using any of these functions, it's important to understand how Cortex AgentIX treats a JSON so you can accurately formulate your queries using the correct syntax.

#### IMPORTANT:

JSON field names are case sensitive, so the key to field pairing must be identical in an XQL query for results to be found. For example, if a field value is "TIMESTAMP" and your query is defined to look for "timestamp", no results will be found.

<json\_path>

Each JSON function includes defining a <json\_path> in both the regular syntax and when using the syntactic sugar format. The <json\_path> argument identifies the data of the JSON object you want to extract using dot-notation. When using the regular syntax, the beginning of the object is represented by a \$. This \$ is not required when using the syntactic sugar format.

Example 85.

If you have the following object:

```
{  
    "a_field" : "This is a_field value",  
    "b_field" : {  
        "c_field" : "This is c_field value"  
    }  
}
```

Then the path using the regular syntax:

\$.a\_field

Returns "This is a\_field value", while the path using the regular syntax:

\$.b\_field.c\_field

Returns "This is c\_field value".

Field in <json\_path> contains characters

In the regular syntax

When using the regular syntax to write your XQL queries and a field in the <json\_path> contains characters, such as a dot (.) or colon (:), the syntax needs to be tweaked slightly to account for the <json\_field>.

For example, when using the `json_extract` function, the previous regular syntax would need to be changed to an updated syntax to account for the field in the <json\_path> containing characters.

Previous regular syntax for the `json_extract` function:

```
json_extract(<json_object_formatted_string>, <json_path>)
```

Updated regular syntax for the `json_extract` function, where the <json\_field> now includes single quotation marks as '<json\_field>':

```
json_extract(<json_object_formatted_string>, "[<json_field>]")
```

For each JSON function, the regular syntax can change slightly, but the "[ '<json\_field>' ]" format is the same. The "[ '<json\_field>' ]" identifies the data you want to extract using dot-notation, where the data extracted is dependent on your syntax.

Example 86.

If you have the following JSON object defined:

```
{"a.b":  
    {"inn":  
        {"one":1}  
    }  
}
```



To extract the data `{"one":1}`, the `"['<json_field>']"` would need to be defined as `"$['a.b'].inn"` for all JSON functions. For example, when using the `json_extract` function, the regular syntax is:

```
json_extract(field_json_1, "$['a.b'].inn")
```

To extract the data `{"inn": {"one":1}}`, the `"['<json_field>']"` would need to be defined as `"$['a.b']"` for all JSON functions. For example, when using the `json_extract` function, the regular syntax is:

```
json_extract(field_json_1, "$['a.b']")
```

Example 87.

If you have the following JSON object defined:

```
{"a.b":  
  {"inn.inn":  
    {"one":1}  
  }  
}
```

To extract the data `{"one":1}`, the `"['<json_field>']"` would need to be defined as `"$['a.b']['inn.inn']"` for all JSON functions. For example, when using the `json_extract` function, the regular syntax is:

```
json_extract(json_field, "$['a.b']['inn.inn'])
```

In the syntactic sugar format

To make it easier for you to write your XQL queries, each JSON function includes an optional syntactic sugar format as opposed to using the regular syntax. When defining the syntactic sugar format and a field in the `<json_path>` contains characters, such as a dot (.) or colon (:), the syntax needs to be tweaked slightly to account for the `<json_field>`.

For example, when using the `json_extract` function, the previous syntactic sugar format would need to be changed to an updated syntax to account for the field in the `<json_path>` containing characters.

Previous syntactic sugar format for the `json_extract` function:

```
<json_object_formatted_string> -> <json_path>{}
```

Updated syntactic sugar format for the `json_extract` function, where the `<json_field>` now includes quotations as "`<json_field>"`:

```
<json_object_formatted_string> -> ["<json_field>"]{}
```

For each JSON function, the syntax of the syntactic sugar format can change slightly, but the `["<json_field>"]` format is the same. The `["<json_field>"]` identifies the data you want to extract using dot-notation, where the data extracted is dependent on your syntax.

Example 88.

If you have the following JSON object defined:

```
{"a.b":  
  {"inn":  
    {"one":1}  
  }  
}
```

To extract the data `{"one":1}`, the `["<json_field>"]` would need to be defined as `["a.b"].inn` for all JSON functions. For example, when using the `json_extract` function, the syntactic sugar format is:

```
json_field -> ["a.b"].inn{}
```

To extract the data `{"inn": {"one":1}}`, the `["<json_field>"]` would need to be defined as `["a.b"]` for all JSON functions. For example, when using the `json_extract` function, the syntactic sugar format is:

```
json_field -> ["a.b"]{}
```

Example 89.

If you have the following `json_object` defined:

```
{"a.b":  
  {"inn.inn":  
    {"one":1}  
  }  
}
```

To extract the data `{"one":1}`, the `["<json_field>"]` would need to be defined as `["a.b"]["inn.inn"]` for all JSON functions. For example, when using the `json_extract` function, the syntactic sugar format is:

```
json_field -> ["a.b"]["inn.inn"]{}
```



### 6.1.7 | How to filter for empty values in the results table

#### Abstract

Learn how to filter for empty values in the results table in Cortex Query Language.

When building a query, you can filter for empty values in the results table, which can include or exclude null or empty strings. In the query syntax, empty strings are represented as " ", while null fields are represented as `null`.

- Exclude null and empty strings using the following syntax:

```
<name of field> != null and <field name> != ""
```

- Include null or empty strings using the following syntax:

```
<name of field> = null or <field name> = ""
```

Example 90.

Below is an example of filtering your endpoint data in the results table to exclude all null values and any empty strings for a user.

```
config timeframe = 90d
| dataset = endpoints
| filter endpoint_status in (CONNECTED, DISCONNECTED)
| filter user != null and user != ""
| fields user, group_names, endpoint_name
```

### 6.1.8 | Understanding string manipulation in XQL

#### Abstract

Learn more about string manipulation in Cortex Query Language (XQL) using double and triple quotes.

When defining string fields in Cortex Query Language (XQL) queries, it's important to understand the various string manipulations available and the syntax required to build effective queries that return the results you're expecting. Cortex Query Language (XQL) uses RE2 for its regular expression implementation.

Cortex AgentiX enables you to use single double quotes ("<text>") or triple double quotes ("""<text>""") when defining your XQL syntax for string manipulation. This specific syntax is used with different stages, functions, and operators, with or without wildcards. Typically, the `alter` and `filter` stages are used with single or triple double quotes, so these stages are used in the examples provided below.

Using single double quotes

Single double quotes ("<text>") include the following functionality:

- Treats the string value literally.
- Wildcards using the asterisk (\*) are processed as XQL wildcards, and match any sequence of characters.
- Escape sequences, such as \n (new line) or \t (tab), are not processed and are treated as plain characters.

Example 91.

"\test\" means to look for \test\

Using triple double quotes

Triple double quotes ("""<text>""") include the following functionality:

- Enables regex-style pattern matching and escape sequence interpretation.
- Escape sequences, such as \n (new line) or \t (tab), are processed.
- Wildcards using the asterisk (\*) are processed as XQL wildcards, and match any sequence of characters.

Example 92.

"""\\test\\"" means to look for \test\

Understanding the results:



- The double backslashes (\\\) at the beginning becomes a single backslash (\) as it's processed as an escaped backslash.
- **test** is interpreted as literal.
- The double backslashes (\\\) at the end becomes a single backslash (\) as it's processed as an escaped backslash.

Query example using alter

When using the **alter** stage, you can use both single ("<text>") and triple ("""<text>"""") double quotes when specifying string values. The difference lies in how special characters and pattern matching are interpreted.

Example 93.

```
config timeframe = 10y
| dataset = test_dataset
| limit 1
| alter test = "\test\
| alter test_triple = """\\test\\"""
| fields test, test_triple
```

#### Understanding the query and results

- **test** field using single double quotes:
  - The field value is "\test\".
  - The output results display \test\ exactly as defined in the field value as no escape sequences are processed.
- **test\_triple** field using triple double quotes:
  - The field value is """\\test\\""".
  - The output results display \ test\ (with a tab between \ and the text est) because:
    - \\: First two backslashes become single backslash \.
    - \t: Interpreted as a tab.
    - est: Is interpreted as literal.
    - \\: Last two backslashes become single backslash \.

Query example using filter

When using the **filter** stage, you can use both single ("<text>") and triple ("""<text>"""") double quotes when specifying string values. The difference lies in how special characters and pattern matching are interpreted.

The examples provided are based on the following data table for a dataset called **test\_dataset**:

_TIME	TEST
Mar 26th 2022 19:26:07	12\t3
May 7th 2023 15:16:00	12 3
Jun 8th 2024 16:56:27	1233
Mar 26th 2024 19:26:07	123
Apr 5th 2024 11:21:02	12\t34563
Apr 9th 2025 13:22:22	1233345
May 9th 2025 13:22:22	12 35897



<b>_TIME</b>	<b>TEST</b>
May 30th 2025 21:45:02	116

Example 94.

```
config timeframe = 10y
| dataset = test_dataset
| filter test = "12\t3*"
| fields test
```

Output results table:

<b>_TIME</b>	<b>TEST</b>
Mar 26th 2022 19:26:07	12\t3
Apr 5th 2024 11:21:02	12\t34563

Explanation of results:

The asterisk (\*) in "12\t3\*" means to process the string field as an XQL wildcard by matching any sequence of characters that begins with 12\t3. In addition, the \t characters are not processed as an escape character, but as plain characters.

Example 95.

```
config timeframe = 10y
| dataset = test_dataset
| filter test = """12\t3*"""
| fields test
```

Output results table:

<b>_TIME</b>	<b>TEST</b>
May 7th 2023 15:16:00	12 3
May 9th 2025 13:22:22	12 35897

Explanation of results:

The \t in """12\t3\*"" is processed as a tab escape character. The asterisk (\*) in """12\t3\*"" means to process the string field as an XQL wildcard by matching any sequence of characters that begins with 12<tab>3.

## 6.2 | Build XQL queries

Abstract

Learn more about how to build Cortex Query Language (XQL) queries using the Query Builder.

To support investigation and analysis, you can search your data by creating queries in the Query Builder. You can create queries with the Cortex Query Language (XQL) or by using the Query Builder templates.

### 6.2.1 | About the Query Builder

Abstract

The Query Builder facilitates threat detection, case expansion, and data analytics for suspected threats.

The Query Builder aids in the detection of threats by allowing you to search for indicators of compromise and suspicious patterns within data sources. It assists in expanding case investigations by identifying related events and entities, such as activities associated with specific user accounts or network lateral



movement. In addition, the Query Builder enables data analytics on suspected threats, helping organizations analyze large volumes of data to identify trends, anomalies, and correlations that may indicate potential security issues. The Query Builder also provides an interactive and visually intuitive way for you to search assets and findings by their relationship types and map them out in real-time.

To support investigation and analysis, you can search all of the data ingested by Cortex AgentiX by creating queries in the Query Builder. You can create queries that investigate leads, expose the root cause of an issue, perform damage assessment, and hunt for threats from your data sources.

Cortex AgentiX provides different options in the Query Builder for creating queries:

- XQL (Build your own queries)

You can use the Cortex Query Language (XQL) to build complex and flexible queries that search specific datasets or presets, or the entire Cortex Data Model (XDM). With XQL Search, you create queries based on stages, functions, and operators. To help you build your queries, Cortex AgentiX provides tools in the interface that provide suggestions as you type, or you can look up predefined queries, common stages and examples. For more information, see How to build XQL queries.

**NOTE:**

Schema changes to datasets may not be reflected in the autocomplete suggestions and definitions as you type in real time the XQL query, and can appear with a slight delay.

**TIP:**

When creating XQL queries, you can:

- Use the up and down arrow keys to navigate through the auto-suggestion commands and definitions.
- Select an auto-suggestion command by pressing either the Enter or Tab key.
- Press Shift+Enter to add a new line, and easily ignore the auto-suggestion output.
- Close the auto-suggestion output by pressing the Esc key.

- Query Builder templates (No XQL knowledge required)

You can use the Query Builder templates to access your data without prior XQL knowledge. The templates include predefined filtering fields and key fieldsets, and can include any field from the XDM schema.

As the templates are also based on XQL, you can also translate your template queries into XQL. With this flexibility, you can enrich the basic queries created by templates for more detailed investigation, or use the templates as a starting point for creating complex queries with full XQL functionality. For more information, see Query Builder templates.

- Graph Search to build queries to search assets, findings, and their contextual data. For more information, see How to build Graph Search queries?.

**TIP:**

If you prefer to use the Query Builder in Legacy mode, switch the toggle in the header. In Legacy mode, the Query Builder searches predefined datasets only. To search the full XDM Data Model, switch to New mode or select XQL Search.

## 6.2.2 | How to build XQL queries

### Abstract

Learn more about how to build XQL queries in the Query Builder.

The Cortex Query Language (XQL) enables you to query data ingested into Cortex AgentiX for rigorous endpoint and network event analysis. To help you create an effective XQL query with the proper syntax, the query field in the user interface provides suggestions and definitions as you type.

XQL forms queries in stages. Each stage performs a specific query operation and is separated by a pipe character (|). Queries require a dataset, or data source, to run against. You can either query the Cortex Data Model (XDM) or you can query specific datasets. In a dataset query, unless otherwise specified, the query runs against the `xdr_data` dataset, which contains all log information that Cortex AgentiX collects from all Cortex product agents, including EDR data, and PAN NGFW data. In XDM queries, you must specify the dataset mapped to the XDM that you want to run your query against.

**IMPORTANT:**

Forensic datasets are not included by default in XQL query results, unless the dataset query is explicitly defined to use a forensic dataset.

Which datasets are mapped to XDM?

The Cortex Query Language (XQL) supports a single Cortex Data Model (XDM), which is a normalized data structure. Datasets are mapped to the XDM in 3 different ways:

1. Automatic default mappings, including the following:



- The `xdr_data` dataset is automatically mapped to the XDM with some data mapping exceptions.
- Next-Generation Firewall (NGFW) network log data are mapped to the XDM from the following datasets:
  - `panw_ngfw_traffic_raw`
  - `panw_ngfw_threat_raw`
  - `panw_ngfw_url_raw`
  - `panw_ngfw_filedata_raw`
  - `panw_ngfw_globalprotect_raw`
  - `panw_ngfw_hipmatch_raw`

2. Out-of-the-box mappings of the datasets as part of the Data Model Rules via the Marketplace. For more information, see Cortex Marketplace.

3. You can create your own mappings by creating your own Data Model Rules. For more information, see Create Data Model Rules.

For more information on the XDM Schema, specifically the fields, fieldsets, fields designated as ENUMS (CONST), and aliases, see the Cortex XSIAM Data Model Schema.

#### XDM query syntax

The basic syntax structure for querying the Cortex Data Model (XDM) is either:

```
datamodel dataset in (<dataset_name>,...) ¦
| <STAGE> ...
| <STAGE> ...
| <STAGE> ...
```

Or

```
datamodel dataset = <dataset_name> ¦
| <STAGE> ...
| <STAGE> ...
| <STAGE> ...
```

In a query using the `datamodel` command, a query runs against the specified datasets, which contain log information ingested by Cortex AgentiX. You can also install Marketplace Content Packs, or map an ingested dataset into the XDM, to query additional datasets.

Adding a wildcard suffix (\*) is supported in the `<dataset_name>`, which matches all datasets that are mapped to the data model and begin with the specified text. For example, `datamodel dataset = xdr*` or `datamodel dataset in (xdr*)`.

When querying the XDM, fields that are not mapped to the XDM are accessible by `<dataset>.<field>`. They can be used at any stage of a `datamodel` query.

When creating XDM queries, auto-suggestions are available, according to the existing XDM fields.

#### Dataset query syntax

In a dataset query, unless otherwise specified, the query runs against the `xdr_data` dataset, which contains all log information that Cortex AgentiX collects from all Cortex product agents, including EDR data, and PAN NGFW data. In a dataset query, if you are running your query against a dataset that has been set as default, there is no need to specify a dataset. Otherwise, specify a dataset in your query. The Dataset Queries lists the available datasets, depending on system configuration.

#### NOTE:

- Users with different dataset permissions can receive different results for the same XQL query.
- An administrator or a user with a predefined user role can create and view queries built with an unknown dataset that currently does not exist in Cortex AgentiX. All other users can only create and view queries built with an existing dataset.
- When you have more than one dataset or lookup, you can change your default dataset by navigating to Settings → Configurations → Data Management → Dataset Management, right-click on the appropriate dataset, and select Set as default. For more information about setting default datasets, see Dataset management.

The basic syntax structure for querying datasets that are not mapped to the XDM is:

```
dataset = <dataset name>
| <stage1> ...
| <stage2> ...
| <stage3> ...
```

Or

```
dataset in (<dataset name>)
| <stage1> ...
| <stage2> ...
| <stage3> ...
```



You can specify a dataset using one of the following formats, which is based on the data retention offerings available in Cortex AgentiX.

- Hot Storage queries use the format `dataset = <dataset name>`. This is the default option.

Example 96.

```
dataset = xdr_data
```

- Cold Storage queries use the format `cold_dataset = <dataset name>`.

Example 97.

```
cold_dataset = xdr_data
```

**NOTE:**

You can build a query that investigates data in both a cold dataset and a hot dataset in the same query. In addition, as the hot storage dataset format is the default option and represents the fully searchable storage, this format is used throughout this guide for investigation and threat hunting. For more information on hot and cold storage, see [Dataset management](#).

When using the hot storage default format, this returns every `xdr_data` record contained in your Cortex AgentiX instance over the time range that you provide to the Query Builder user interface. This can be a large amount of data, which may take a long time to retrieve. You can use a `limit` stage to specify how many records you want to retrieve.

There is no practical limit to the number of stages that you can specify. See [Stages](#) for information on all the supported stages.

In the `xdr_data` dataset, every user `_uid` included in the raw data for network, authentication, and login events has an equivalent normalized user `_uid` associated with it that displays the user information in the following standardized format:

```
<company domain>\<username>
```

For example, the `login_data` `_uid` has the `login_data_dst_normalized_user` `_uid` to display the content in the standardized format. To ensure the most accurate results, we recommend that you use these `normalized_user` `_uids` when building your queries.

**Additional components**

XQL queries can contain different components, such as functions and stages, depending on the type of query you want to build. For a complete list of the syntax options available with example queries, see [Stages and Functions](#).

#### 6.2.2.1 | Get started with XQL queries

**Abstract**

Learn more about some important information before getting started with XQL queries.

Before you begin running XQL queries, consider the following information:

- Use the interface to help you build queries

Cortex AgentiX offers features in the XQL search interface to help you build queries. For more information, see [Useful XQL user interface features](#).

- Mitigate long-running queries

Querying the XDM enables searching of Cortex AgentiX's extensive data. We recommend that you use filters to streamline your queries. For more information, see [XQL Query best practices](#).

- Understand query defaults and limitations

Before you run a query, review this list to better understand query behavior and results. For more information, see [Expected results when querying fields](#).

- Translate Splunk queries to XQL

If you have existing Splunk queries, you can translate them to XQL. For more information, see [Translate to XQL](#).

**TIP:**

If you are new to creating queries, you can also try our simple search templates, which can help you get started in understanding how queries work. See [Query Builder templates](#).

#### 6.2.2.2 | Useful XQL user interface features

**Abstract**

Learn about useful XQL query features in the user interface.

The user interface contains several useful features for querying data, and for viewing results:



- XQL query: The XQL query field is where you define the parameters of your query. To help you create an effective XQL query, the search field provides suggestions and definitions as you type.

**NOTE:**

Schema changes to datasets may not be reflected in the autocomplete suggestions and definitions as you type in real time the XQL query and can appear with a slight delay.

**TIP:**

When creating XQL queries, you can:

- Use the up and down arrow keys to navigate through the auto-suggestion command suggestions and definitions.
- Select an auto-suggestion command by pressing either the Enter or Tab key.
- Press Shift+Enter to add a new line, and easily ignore the auto-suggestion output.
- Close the auto-suggestion output by pressing the Esc key.
- Translate to XQL: Converts your existing Splunk queries to the XQL syntax. When you enable Translate to XQL, both an SPL query field and an XQL query field are displayed. You can easily add a Splunk query, which is converted automatically into XQL in the XQL query field. This option is disabled by default.
- Query Results: After you create and run an XQL query, you can view, filter, and visualize your Query Results.
- XQL Helper: Describes common stage commands and provides examples that you can use to build a query.
- Query Library: Contains common, predefined queries that you can use or modify to your liking. In addition, there is a personal query library for saving and managing your own queries so that you can share with others, and queries can be shared with you. For more information, see Manage your personal query library.
- Schema: Contains schema information for every field found in the result set. This information includes the field name, data type, descriptive text (if available), and the dataset that contains the field.
  - For dataset queries, it contains the list of all the fields of all the datasets that were involved in the query.
  - For data model queries, it contains the list of all the data model fields.

#### 6.2.2.3 | XQL Query best practices

##### Abstract

Learn about best practices for streamlining XQL queries.

Cortex AgentX includes built-in mechanisms for mitigating long-running queries, such as default limits for the maximum number of allowed issues, and for the maximum number of returned rows. Only specified mapped datasets are searched when querying by the Cortex Data Model (XDM) to use system resources and time more efficiently. The following suggestions can help you to streamline your queries:

- Add a smaller limit to queries by using a `limit` stage.

To help reduce the Cortex Query Language (XQL) response time, the default results for an XDM query or an XQL dataset query is limited to 1000, when no limit is explicitly stated in the query. This applies to basic queries with no stages except the `fields` stage. This default limit does not apply to widgets, Correlation Rules, public APIs, saved queries, or scheduled queries, where the limit is a maximum of 1,000,000 results. Queries based on legacy templates are limited to 10,000 results. Adding a smaller limit can greatly reduce the response time.

Example 98.

```
datamodel dataset = microsoft_windows_raw
| fields *host*
| limit 100
```

- Use a small time frame for queries by specifying the specific date and time in the Timeframe, such as selecting Relative time and defining Last 30 Minutes, instead of picking the nearest larger option available or defining an extended time period.
- Use filters that exclude data, along with other possible filters.
- Select the specific fields that you would like to see in the query results.

#### 6.2.2.4 | Expected results when querying fields

##### Abstract

Learn what to expect in the query results when querying fields.



The following are returned when querying fields:

- If specific fields are stated in the fields stage, those exact fields will be returned.
- If no fields are stated in the query, the `xdm_core` fieldset will be returned.
- Unmapped fields are treated as NULL. An unmapped field is an `xdm` field that hasn't been mapped from the relevant datasets using a Data Model Rule.
- By default, the `_time` system field will be added to all data model queries. Yet, the `_time` system field will not be added to queries that contain the `comp` stage.
- For dataset queries, all current system fields will be returned, even if they are not stated in the query.
- For UNION between XDM and dataset, each part of the UNION will return its own fields.
- Each new column in the result set created by the alter stage will be added as the last column. You can specify a different column order by modifying the field order in the fields stage of the query.
- Each new column in the result set created by the comp stage will be added as the last column. Other fields that are not in the `group by` / `calculated` column will be removed from the result set, including the core fields and `_time` system field.
- When no limit is explicitly stated in a `datamodel` query, a maximum of 1000 results are returned (default). When this limit is applied to results using the `limit` stage, it will be indicated in the user interface.

#### 6.2.2.5 | Create XQL query

##### Abstract

Learn how to create queries using the Cortex Query Language (XQL).

Review the following topics:

- How to build XQL queries

Build Cortex Query Language (XQL) queries to analyze raw log data stored in Cortex AgentIX. You can query the Cortex Data Model (XDM) or datasets using specific syntax.

##### How to create a XDM query

1. From Cortex AgentIX, select Investigation & Response → Search → Query Builder.
2. Click XQL.
3. (*Optional*) Change the default time period against which to run your query from the time picker at the top right of the window. You can select the required time period from any of the following options available:
  - Preset time ranges easily available to select from, such as 24 hours and 30 days.
  - Recently used selections from your previous queries.
  - Relative time: Define the time frame as the last <number> minutes, days, or hours by setting the number.
  - Calendar: Create a customized time period by selecting the date range from the calendar and the specific Start Time and End Time.

##### NOTE:

- Whenever the time period is changed in the query window, the `config timeframe` is automatically set to the time period defined for the entire query, including queries that are part of the `join` stage. Yet, this won't be visible as part of the query. Only if you manually type in the `config timeframe` will this be seen in the query.
- These time picker options are available in XQL queries when using the Query Builder, XQL Widgets, and when defining XQL Widgets in Reports and Dashboards.

4. (*Optional*) To translate Splunk queries to XQL queries, enable Translate to XQL. If you choose to use this feature, enter your Splunk query in the Splunk field, click the arrow icon to convert to XQL, and then go to Step 6.
5. Create your query by typing in the query field. Relevant commands, their definitions, and operators are suggested as you type.

##### TIP:

When creating XQL queries, you can:

- Use the up and down arrow keys to navigate through the auto-suggestion command suggestions and definitions.
- Select an auto-suggestion command by pressing either the Enter or Tab key.
- Press Shift+Enter to add a new line, and easily ignore the auto-suggestion output.
- Close the auto-suggestion output by pressing the Esc key.



- a. Specify the datasets to run your query against by typing either `datamodel dataset = <dataset name>...` or `datamodel dataset in (<dataset name>, ...)`. For example:

```
datamodel dataset in (amazon_aws_raw)
```

**NOTE:**

While `datamodel dataset=*` is supported in the query, we recommend that you specify specific datasets for quicker and more efficient results.

- b. Press Enter, and then type the pipe character (|). Select a stage, and complete the stage syntax using the suggested options.

- c. Continue adding stages until your query is complete. For example:

```
datamodel dataset in (amazon_aws_raw)
| filter xdm.source.ipv4 = "10.9.165.1"
| fields xdm.source.ipv4, xdm.source.port
| limit 100
```

6. Choose when to run your query:

- Run the query immediately.
- Run the query by the specified date and time, or on a specific date, by selecting the calendar icon ().

7. (Optional) The Save As options save your query for future use:

- Correlation Rule: When compatible, saves the query as a Correlation Rule. For more information, see What's a correlation rule?.
- Query to Library: Saves the query to your personal query library. For more information, see Manage your personal query library.
- Widget to Library: For more information, see Create custom XQL widgets.

**TIP:**

While the query is running, you can navigate away from the page. A notification is sent when the query has finished. You can also Cancel the query or run a new query, where you have the option to Run only new query (cancel previous) or Run both queries.

How to create a dataset query

1. From Cortex AgentIX, select Investigation & Response  Search 

2. Click XQL.

3. (Optional) Change the default time period against which to run your query from the time picker at the top right of the window. You can select the required Timeframe from any of the following options available:

- Preset time ranges easily available to select from, such as 24 hours and 30 days.
- Recently used selections from your previous queries.
- Relative time: Define the time frame as the last <number> minutes, days, or hours by setting the number.
- Calendar: Create a customized time period by selecting the date range from the calendar and the specific Start Time and End Time.

**NOTE:**

- Whenever the time period is changed in the query window, the `config timeframe` is automatically set to the time period defined for the entire query, including queries that are part of the join stage. Yet, this won't be visible as part of the query. Only if you manually type in the `config timeframe` will this be seen in the query.
- These time picker options are available in XQL queries when using the Query Builder, XQL Widgets, and when defining XQL Widgets in Reports and Dashboards.

4. (Optional) To translate Splunk queries to XQL queries, enable Translate to XQL. If you choose to use this feature, enter your Splunk query in the Splunk field, click the arrow icon () to convert to XQL, and then go to Step 6.

5. Create your query by typing in the query field. Relevant commands, their definitions, and operators are suggested as you type.

**TIP:**

When creating XQL queries, you can:

- Use the up and down arrow keys to navigate through the auto-suggestion command suggestions and definitions.
- Select an auto-suggestion command by pressing either the Enter or Tab key.
- Press Shift+Enter to add a new line, and easily ignore the auto-suggestion output.
- Close the auto-suggestion output by pressing the Esc key.

- a. (Optional) Specify a dataset.

You only need to specify a dataset if you are running your query against a dataset that you have not set as default. Otherwise, the query runs against the `xdr_data` dataset. For more information, see How to build XQL queries.



Example 99.

```
dataset = xdr_data
```

b. Press Enter, and then type the pipe character (|). Select a command, and complete the command using the suggested options.

c. Continue adding stages until your query is complete.

Example 100.

```
dataset = xdr_data
| filter agent_os_type = ENUM.AGENT_OS_MAC
| limit 250
```

6. Choose when to run your query:

- Run the query immediately.
- Run the query by the specified date and time, or on a specific date, by selecting the calendar icon (📅).

7. (Optional) The Save As options save your query for future use:

- BIOC Rule: When compatible, saves the query as a BIOC rule. The XQL query must contain a filter for the event\_type field.
- Correlation Rule: When compatible, saves the query as a Correlation Rule. For more information, see What's a correlation rule?.
- Query to Library: Saves the query to your personal query library. For more information, see Manage your personal query library.
- Widget to Library: For more information, see Create custom XQL widgets.

**TIP:**

While the query is running, you can navigate away from the page. A notification is sent when the query has finished. You can also Cancel the query or run a new query, where you have the option to Run only new query (cancel previous) or Run both queries.

#### 6.2.2.6 | Review XQL query results

##### Abstract

Learn more about reviewing the results returned from an XQL query.

Review the following topics:

- How to build XQL queries
- Create XQL query

The results of a Cortex Query Language (XQL) query are displayed in the Query Results tab.

**NOTE:**

It's also possible to graph the results displayed. For more information, see Graph query results.

##### Real-time query results

Cortex AgentX displays partial results for queries run in the Query Builder as they are received, subject to the limitations below. In a long-running query, viewing the initial findings enables you to refine, validate, or stop the query.

The partial results are displayed only in the Table tab. The results are added to the table as they are received in real time. The incremental query results aren't ordered, so they may not be in sequence.

**NOTE:**

Results are received incrementally for the first 100K records, or up to 100MB worth of records, whichever comes first. After that, the next update is when the query has finished running completely.

**NOTE:**



- Real time query results are available only in the Query Builder and in free text query.
- Real time results are displayed only for queries run on hot datasets.
- The Sort option is available only after all the data is retrieved.
- When you formulate complex queries, the results will be displayed when the query has finished running completely, and not in real time. Some of the clauses that are included in this restriction are:
  - JOIN - incremental results are supported only when the secondary dataset is smaller in size
  - SORT
  - COMP
  - WINDOWCOMP
  - TOP

Understanding the options available to investigate results

Use the following options in the Query Results tab to investigate your query results:

Option	Use
Table tab	<p>Displays results in rows and columns according to the entity fields. Columns can be filtered, using their filter icons.</p> <p>More options (⋮) displays table layout options, which are divided into different sections:</p> <ul style="list-style-type: none"> <li>• In the Appearance section, you can Show line breaks for any text field in the Query Results. By default, the text in these fields are wrapped unless the Show line breaks option is selected. In addition, you can change the way rows and columns are displayed.</li> <li>• In the Log Format section, you can change the way that logs are displayed:           <ul style="list-style-type: none"> <li>◦ RAW: Raw format of the entity in the database.</li> <li>◦ JSON: Condensed JSON format with key value distinctions. NULL values are not displayed.</li> <li>◦ TREE: Dynamic view of the JSON hierarchy with the option to collapse and expand the different hierarchies.</li> </ul> </li> <li>• In the Search column section, you can find a specific column; enable or disable display of columns using the checkboxes.</li> </ul> <p>Show and hide rows according to a specific field in a specific event: select a cell, right-click it, and then select either Show rows with ⓘ or Hide rows with ⓘ</p>
Graph tab	Use the Chart Editor to visualize the query results.
Advanced tab	<p>Displays results in a table format which aggregates the entity fields into one column. You can change the layout, decide whether to Show line breaks for any text field in the results table, and change the log format from the ⋮ menu.</p> <p>Select Show more to pivot an Expanded View of the event results that include NULL values. You can toggle between the JSON and Tree views, search, and Copy to clipboard.</p>
Export to File	<p>Exports the results to a TSV (tab-separated values) file.</p> <ul style="list-style-type: none"> <li>• More options (⋮) works in a similar way to how it works on the Table tab.</li> <li>• Show more in the bottom left corner of each row opens the Expanded View of the event results that also include NULL values. Here, you can toggle between the JSON and Tree views, search, and Copy to clipboard.</li> <li>• Log format options change the way that logs are displayed:           <ul style="list-style-type: none"> <li>◦ RAW: Raw format of the entity in the database.</li> <li>◦ JSON: Condensed JSON format with key value distinctions. NULL values are not displayed.</li> <li>◦ TREE: Dynamic view of the JSON hierarchy with the option to collapse and expand the different hierarchies.</li> </ul> </li> </ul>
Refresh	Refreshes the query results.



Option	Use
Free text search	Searches the query results for text that you specify in the free text search. Click the Free text search icon to reveal or hide the free text search field.
Filter	Enables you to filter a particular field in the interface that is displayed to specify your filter criteria. For integer, boolean, and timestamp (such as <code>_time</code> ) fields, we recommend that you use the Filter instead of the Free text search, in order to retrieve the most accurate query results.
Fields menu	Filters query results. To quickly set a filter, Cortex AgentiX displays the top ten results from which you can choose to build your filter. This option is only available in the Table and Advanced tabs. From within the Fields menu, click on any field (excluding JSON and array fields) to see a histogram of all the values found in the result set for that field. This histogram includes: <ul style="list-style-type: none"> <li>• A count of the total number of times a value was found in the result set.</li> <li>• The value's frequency as a percentage of the total number of values found for the field.</li> <li>• A bar chart showing the value's frequency.</li> </ul> <p><b>NOTE:</b></p> <p>In order for Cortex AgentiX to provide a histogram for a field, the field must not contain an array or a JSON object.</p>

#### Available options for saving results

The Save As options save your query for future use:

- 
- Correlation Rule: When compatible, saves the query as a Correlation Rule. For more information, see [What's a correlation rule?](#).
- Query to Library: Saves the query to your personal query library. For more information, see [personal query library](#).
- Widget to Library: For more information, see [???](#).

#### Investigating results in the Causality View or Timeline View

You can continue investigating the query results in the Causality View or Timeline by right-clicking the event and selecting the desired view. This option is available for the following types of events:

- Process (except for those with an event sub-type of termination)
- Network
- File
- Registry
- Injection
- Load image
- System calls
- Event logs for Windows
- System authentication logs for Linux

For network stories, you can pivot to the Causality View only. For cloud Cortex AgentiX events and Cloud Audit Logs, you can only pivot to the Cloud Causality View, while software-as-a-service (SaaS) related issues for audit stories, such as Office 365 audit logs and normalized logs, you can only pivot to the SaaS Causality View.

#### Add file path to Malware Profile allowed list

Add a file path to your existing Malware Profile allowed list by right-clicking a `<path>` field, such as `target_process_path`, and select Add `<path type>` to malware profile allow list.



## 6.2.2.7 | Translate to XQL

### Abstract

Learn how to translate your Splunk queries to XQL queries in Cortex AgentiX.

To help you easily convert your existing Splunk queries to the Cortex Query Language (XQL) syntax, Cortex AgentiX includes a toggle called Translate to XQL in the query field in the user interface. When building your XQL query and this option is selected, both a SPL query field and XQL query field are displayed, so you can easily add a Splunk query, which is converted to XQL in the XQL query field. This option is disabled by default, so only the XQL query field is displayed.

#### **IMPORTANT:**

This feature is still in a Beta state and you will find that not all Splunk queries can be converted to XQL. This feature will be improved upon in the upcoming releases to support greater Splunk query translations to XQL.

### Supported functions in Splunk

The following table details the supported functions in Splunk that can be converted to XQL in Cortex AgentiX with an example of a Splunk query and the resulting XQL query. In each of these examples, the `xdr_data` dataset is used.

Splunk Function/Stage	Splunk Query Example	Resulting XQL Query
avg	<code>index=xdr_data   stats avg(dst_association_strength)</code>	<code>dataset in (xdr_data)   comp avg(dst_association_strength)</code>
bin	<code>index = xdr_data   bin _time span=5m</code>	<code>dataset in (xdr_data)   bin _time span=5m</code>
coalesce	<code>index= xdr_data   eval product_or_vendor_not_null=coalesce(_product, _vendor )</code>	<code>dataset in (xdr_data)   alter product_or_vendor_not_null</code>
count	<code>index=xdr_data   stats count(_product) BY _time</code>	<code>dataset in (xdr_data)   comp count(_product) by _time</code>
ctime	<code>index=xdr_data   convert ctime(field) as field</code>	<code>dataset in (xdr_data)   alter field = format_timestamp(</code>
earliest	<code>index = xdr_data earliest=24d</code>	<code>dataset in (xdr_data)   filter _time &gt;= to_timestamp(ad</code>
eval	<code>index=xdr_data   eval field = "test"</code>	<code>dataset in (xdr_data)   alter field = "test"</code>
fillnull	<code>index=xdr_data   fillnull value = "missing ipv6" agent_ip_addresses_v6</code>	<code>dataset in (xdr_data)   replacenull agent_ip_addresses_</code>
floor	<code>index=xdr_data   eval floor_test = floor(1.9)</code>	<code>dataset in (xdr_data)   alter floor_test = floor(1.9)</code>
iplocation	<code>index=xdr_data   inputlookup append=true my_lookup.csv</code>	<code>dataset in (xdr_data)   union (dataset=my_lookup   limi</code>
iplocation	<code>index = xdr_data   inputlookup agent_ip_addresses</code>	<code>dataset in (xdr_data)   iploc agent_ip_addresses loc_co</code>
isnotnull	<code>index=xdr_data   eval x = isnotnull(agent_hostname)</code>	<code>dataset in (xdr_data)\n   alter x = if(agent_hostname !</code>



Splunk Function/Stage	Splunk Query Example	Resulting XQL C
isnull	index=xdr_data   eval x = isnull(agent_hostname)	dataset in (xdr_data)\n   alter x = if(agent_hostname = null, 1, 0)
json_extract	index= xdr_data   eval London=json_extract(dfe_labels,"dfe_labels{0}")	dataset in (xdr_data)   alter London = dfe_labels -> dfelabels[0]
join	join agent_hostname [index = xdr_data]	join type=left conflict_strategy=right (dataset in (xdr_data)) by agent_hostname
latest	index = xdr_data latest=-24d	dataset in (xdr_data)   filter _time <= to_timestamp(add(to_epoch(date_floor(current_time()), "days"), -24))
len	index = xdr_data   where uri != null   eval length = len(agent_ip_address)	dataset in (xdr_data)   filter agent_ip_addresses != null   eval length = len(agent_ip_addresses)
ltrim(<str>, <trim_chars>)	index=xdr_data   eval trimed_agent=ltrim("agent_hostname", "agent_")	dataset in (xdr_data)   alter trimed_agent = ltrim("agent_hostname", "agent_")
lower	index = xdr_data   eval field = lower("TEST")	dataset in (xdr_data)   alter field = lowercase("TEST")
max	index =xdr_data   stats max(action_file_size) by _product	dataset in (xdr_data)   comp max(action_file_size) by _product
md5	index=xdr_data   eval md5_test = md5("test")	dataset in (xdr_data)   alter md5_test = md5("test")
median	index = xdr_data   stats median(actor_process_file_size) by _time	dataset in (xdr_data)   comp median(actor_process_file_size) by _time
min	index =xdr_data   stats min(action_file_size) by _product	dataset in (xdr_data)   comp min(action_file_size) by _product
mvcount	index = xdr_data   where http_data != null   eval http_data_array_length = mvcount(http_data)	dataset in (xdr_data)   filter http_data != null   alter http_data_array_length = mvcount(http_data)
mvdedup	index = xdr_data   eval s=mvdedup(action_app_id_transitions)	dataset in (xdr_data)   alter s = arraydistinct(action_app_id_transitions)
mvexpand	index = xdr_data   mvexpand dfe_labels limit = 100	dataset in (xdr_data)   arrayexpand dfe_labels limit 100
mvfilter	index = xdr_data   eval x = mvfilter(isnull(dfe_labels))	dataset in (xdr_data)   alter x = arrayfilter(dfe_labels, isnull)
mvindex	index=xdr_data   eval field = mvindex(action_app_id_transitions, 0)	dataset in (xdr_data)   alter field = arrayindex(action_app_id_transitions, 0)



Splunk Function/Stage	Splunk Query Example	Resulting XQL C
mvjoin	index=xdr_data   eval n=mvjoin(action_app_id_transitions, ";")	dataset in (xdr_data)   alter n = arraystring(action_ap
pow	index=xdr_data   eval pow_test = pow(2, 3)	dataset in (xdr_data)   alter pow_test = pow(2, 3)
relative_time(X,Y)	<ul style="list-style-type: none"> <li>index ="xdr_data"   where _time &gt; relative_time(now(),"-7d@d")</li> <li>index ="xdr_data"   where _time &gt; relative_time(now(),"+7d@d")</li> </ul>	<ul style="list-style-type: none"> <li>dataset in (xdr_data)   filter _time &gt; to_timestamp(add(to_epoch(date_floor(current_time())), -7 * 24 * 3600))</li> <li>dataset in (xdr_data)   filter _time &gt; to_timestamp(add(to_epoch(date_floor(current_time())), 7 * 24 * 3600))</li> </ul>
replace	index= xdr_data   eval description = replace(agent_hostname, "\("."NEW")	dataset in (xdr_data)   alter description = replace(age
rex	index=xdr_data action_local_ip!="0.0.0.0"   rex field=action_local_ip "(? <src_ip>\d+\.\d+\.\d+\.\d+)"   where src_ip != ""   table action_local_ip src_ip</src_ip>	dataset in (xdr_data)   filter (action_local_ip != "0.0.0.0") arrayindex(regexextract(action_local_ip, "(\d+\.\d+\.\d+\.\d+)")) as action_local_ip, src_ip
round	index=xdr_data   eval round_num = round(3.5)	dataset in (xdr_data)   alter round_num = round(3.5)
rtrim	index=xdr_data   eval trimed_hostname=rtrim("agent_hostname", "hostname")	dataset in (xdr_data)   alter trimed_hostname = rtrim("agent_hostname", "hostname")
search	index = xdr_data   eval ip="192.0.2.56"   search ip="192.0.2.0/24"	dataset in (xdr_data)   alter ip = "192.0.2.56"   filter ip="192.0.2.0/24"
sha256	index = xdr_data   eval sha256_test = sha256("test")	dataset in (xdr_data)   alter sha256_test = sha256("tes
sort (ascending order)	index = xdr_data   sort action_file_size	dataset in (xdr_data)   sort asc action_file_size   lim
sort (descending order)	index = xdr_data   sort -action_file_size	dataset in (xdr_data)   sort desc action_file_size   li
spath	index = xdr_data   spath output=myfield input=action_network_http path=headers.User-Agent	dataset in (xdr_data)   alter myfield = json_extract(ac
split	index = xdr_data   where mac != null   eval split_mac_address = split(mac, ":")	dataset in (xdr_data)\n   filter mac != null\n   alter
stats	index=xdr_data   stats count(event_type) by _time	dataset in (xdr_data)   comp count(event_type) by _time



Splunk Function/Stage	Splunk Query Example	Resulting XQL
stats dc	index = xdr_data   stats dc(_product) BY _time	dataset in (xdr_data)   comp count_distinct(_product) by _time
strcat	index=xdr_data   strcat story_id "/" http_req_before_method comboIP	dataset in (xdr_data)   alter comboIP=concat(if(story_id!=null,story_id,""),"/",if(ht)
sum	index=xdr_data   where action_file_size != null   stats sum(action_file_size) by _time	dataset in (xdr_data)   filter action_file_size != null   sum(action_file_size) by _time
table	index = xdr_data   table _time, agent_hostname, agent_ip_addresses, _product	dataset in (xdr_data)   fields _time, agent_hostname, agent_ip_addresses, _product
tonumber	index=xdr_data   eval tonumber_test = tonumber("90210")	dataset in (xdr_data)   alter tonumber_test = to_number(tonumber("90210"))
top	<p>The following Splunk functions can be translated to XQL:</p> <ul style="list-style-type: none"> <li>• limit</li> <pre>index = xdr_data   where action_app_id_risk &gt; 0   top limit=20 action_app_id_risk</pre> <li>• countfield</li> <pre>index = xdr_data   top countfield=count_agent_hostname agent_hostname by _time</pre> <li>• showcount</li> <pre>index = xdr_data   where action_app_id_risk &gt; 0   top 3 showcount=t action_app_id_risk</pre> <li>• showperc</li> <pre>index = xdr_data   where action_app_id_risk &gt; 0   top 3 showperc=t action_app_id_risk</pre> <li>• percentfield</li> <pre>index = xdr_data   top percentfield=agent_hostname_percentage agent_hostname by _time</pre> </ul>	<ul style="list-style-type: none"> <li>• limit</li> <pre>dataset in (xdr_data)   filter action_app_id_risk &gt; 0   top_percent as percent</pre> <li>• countfield</li> <pre>dataset in (xdr_data)   top 10 agent_hostname by _time   countfield=agent_hostname   percent</pre> <li>• showcount</li> <pre>dataset in (xdr_data)   filter action_app_id_risk &gt; 0   top_percent as percent</pre> <li>• showperc</li> <pre>dataset in (xdr_data)   filter action_app_id_risk &gt; 0   top_percent as percent</pre> <li>• percentfield</li> <pre>dataset in (xdr_data)   top 10 agent_hostname by _time   percentfield=agent_hostname_percentage</pre> </ul>
upper	index=xdr_data   eval field = upper("test")	dataset in (xdr_data)   alter field = uppercase("test")
var	index=xdr_data   stats var (event_type) by _time	dataset in (xdr_data)   comp var(event_type) by _time

How to translate a Splunk query to XQL syntax

1. Select Investigation & Response âœ Search âœ Query Builder âœ XQL.
2. Toggle to Translate to XQL, where both a SPL query field and XQL query field are displayed.
3. Add your Splunk query to the SPL query field.



4. Click the arrow ().

The XQL query field displays the equivalent Splunk query using the XQL syntax.

You can now decide what to do with this query based on the instructions explained in Create XQL query.

#### 6.2.2.8 | Graph query results

##### Abstract

Cortex AgentIX enables you to generate helpful visualizations of your XQL query results.

To help you better understand your Cortex Query Language (XQL) query results and share your insights with others, Cortex AgentIX enables you to generate graphs and outputs of your query data directly from query results page.

1. Select Investigation & Response → Search → Query Builder → XQL.

2. Run an XQL query.

Example 101.

Enter the following query:

```
dataset = xdr_data
| fields action_total_upload, _time
| limit 10
```

The query returns the `action_total_upload`, a number field, and `_time`, a string field, for up to 10 results.

3. In the Query Results section, to graph the results either:

Use Chart Editor

Navigate to Query Results →  Chart Editor to manually build and view the graph using the selected graph parameters:

- Main
  - Graph Type: Type of graphs and output options available: Area, Bubble, Column, Funnel, Gauge, Line, Map, Pie, Scatter, Single Value, or Word Cloud.
- NOTE:

To display the result of as a time duration, choose the graph type Single Value and enable Show as Time. You can then select the Time Unit (millisecond, second, minute, or hour) and the Display format.
- Subtype and Layout: Depending on the selected type of graph, choose from the available display options.
- Header: Title your graph.
- Show Callouts: Display numeric values on the graph.
- Data
  - X-axis: Select a field with a string value.
  - Y-axis: Select a field with a numeric value.
  - (Optional) Series: For an area, bubble, column, line, map, or scatter chart, you can specify a field (column) to group chart results based on y-axis values. This option is only displayed when one of the supported graph types are selected, and a single y-axis value is selected.
- Depending on the selected type of graph, customize the Color, Font, and Legend.

Use XQL query

Enter the visualization parameters in the XQL query section.

You can express any chart preferences in XQL. This is helpful when you want to save your chart preferences in a query and generate a chart every time that you run it. To define the parameters, either:



- Define the following query:

Example 102.

```
dataset = xdri_data
| view graph type = column header = "Test 1" xaxis = _time yaxis = action_total_upload series = _vendor
```

- Select ADD TO QUERY to insert your chart preferences into the query itself.

#### 4. (Optional) Create a custom widget.

To easily track your query results, you can create custom widgets based on the query results. The custom widgets you create can be used in your custom dashboards and reports. For more information, see [Create custom XQL widgets](#).

Select Save to Widget Library to pivot to the Widget Library and generate a custom widget based on the query results.

### 6.2.3 | Query Builder templates

Abstract

Use Query Builder templates to query your data sets without using the Cortex Query Language.

You can use the Query Builder templates to create effective queries without using the Cortex Query Language (XQL).

From the Query Builder, you can select the following templates:

- Basic: Search by IP address, host name, user name, and domain.
- Free text: Search for a free text string.

The templates are set up with predefined filtering fields and fieldsets that are specific to the template type. You can specify values for the default fields and add any other required fields to refine and adapt your search. The Query Builder templates support any filtering fields from the Cortex Data Model (XDM) schema.

#### TIP:

To get started with queries, you can run an empty template query with no values specified. The query results will include all of the fields in the template specific fieldset. Based on the query results, you can run subsequent queries to narrow down your search.

#### 6.2.3.1 | Get started with Query Builder templates

Abstract

Information to help you get started with Query Builder templates.

Before you start running queries with Query Builder templates, consider the following information:

- **Learn about the templates:** Although the templates don't require XQL knowledge, they do require knowledge of operators and other factors. Understanding how the templates work will help you to build effective queries. For more information, see [Considerations for using Query Builder templates](#).
- **Look up field and alias descriptions:** The templates are based on the fields and aliases in the Cortex Data Model (XDM). If you want more information about a field or alias, see the [Cortex XSIAM Data Model Schema Guide](#).

#### 6.2.3.2 | Considerations for using Query Builder templates

Abstract

Learn more about using filtering fields and operators in Query Builder templates.

The following sections provide information and considerations for using Query Builder templates.

##### General considerations

The following general considerations apply to Query Builder templates:



- The templates run on the following datasets by default:
  - Basic, Identity, Endpoint, and Network templates: `xdr_data`
  - Cloud template: `cloud_audit_logs`

It is also possible to run the templates on all datasets.

- The query uses an AND operator between the filtering fields.
- Separate multiple values with pipes and do not add spaces between the value and the pipe.
- Some of the filtering fields are aliases and therefore search all fields that are associated with the alias.
- Fields with dropdown options support ENUMs and free text values.
- In IP address fields, you can also specify subnets.
- The asterisk (\*) wildcard is supported, except in subnet values.
- You cannot remove the predefined fields, but you can leave them blank.
- When filtering integer and float fields, you can only specify two operators from the four available options.

#### `= (equal to) and != (not equal to)` operators

Filtering fields support the `= (equal to)` and `!= (not equal to)` operators, and you can specify both operators for the same field. The following conditions apply to these operators:

- If you specify multiple values for a field with the `=` operator, the OR operator is applied. For example, `User Name = aaa|bbb` searches for instances of user name equal to aaa OR bbb.
- If you specify multiple values for a field with the `!=` operator, the AND operator is applied. For example, `User Name != aaa|bbb` searches for instances of user name not equal to aaa AND bbb.
- If you specify both operators (`=` and `!=`) for the same field, the AND operator is applied. For example, `COUNTRY = Empty values AND COUNTRY != USA`.

#### `>= (greater than and equal) and <= (less than and equal)` operators

Filtering fields support the `>= (greater than and equal)` and `<= (less than and equal)` operators, and you can specify both operators for the same field. The following conditions apply to these operators:

- Cortex AgentiX supports using these operators for integer and float fields.
- Empty values are not supported with these operators.

#### Include and exclude empty values

You can use the Empty values field to include or exclude fields with empty values and strings. In the search results, some fields might return empty values. This occurs if no data is mapped to a field. The following conditions apply to the Empty values field:

- If you specify `=` and select Empty values, the query includes fields with empty values with an OR operator.  
For example, `_vendor = aaa OR _vendor = Empty values` searches the `_vendor` field for any instances of aaa or empty values.
- If you specify `!=` and select Empty values, the query excludes fields with empty values with an AND operator.  
For example, `_vendor != aaa AND _vendor != Empty values` searches the `_vendor` field for values that are not equal to aaa AND do not contain empty values.
- If you specify `!=` and select Empty values for an alias, you might not receive any results. The query searches all of the fields associated with the alias for non-empty values. If any of the associated fields contain empty values, no results are returned.

For example, `User Name != aaa AND User Name != Empty values` searches the User Name alias fields for values that are not equal to aaa AND empty values. If the query finds either aaa or empty values in any of the alias fields, no results are returned.

### 6.2.3.3 | Create a query from a template

#### Abstract

Explains how to run queries with Query Builder templates that do not require prior Cortex Query Language (XQL) knowledge.

You can use the Query Builder templates to create effective queries without using the Cortex Query Language (XQL).



Review the following topics:

- Query Builder templates
- Get started with Query Builder templates
- Considerations for using Query Builder templates

How to create a query from a Query Builder template

1. Select Investigation & Response → Search → Query Builder.

2. In the Query Builder, select the template that you want to use.

If you want to use the Free Text Search template, see Run a free text query.

3. (Optional) Change the Run on option (upper-right corner) that controls the datasets configured to run with the template. The templates are automatically configured to run on default datasets or you can choose to run them on all datasets. The templates run on the following datasets by default:

- Basic, Identity, Endpoint, and Network templates: `xdr_data`
- Cloud template: `cloud_audit_logs`

4. Enter values for any of the predefined fields and specify whether to include Empty values in the query.

Guidelines

- The query uses an AND operator between the filtering fields.
- Separate multiple values with pipes and do not add spaces between the value and the pipe.
- Some of the filtering fields are aliases and therefore search all fields that are associated with the alias.
- You can run an empty template with no values specified. The query results will show data from all of the fields in the template specific fieldset.

For more information about using the filtering fields, operators, and including Empty values, see Considerations for using Query Builder templates.

5. (Optional) Click Add Field and select the additional filtering fields or aliases to include in the query."

**NOTE:**

- Field names and aliases are listed without their prefix, for example `xdm.SOURCE.USER.USERNAME` is listed as `SOURCE.USER.USERNAME` and `XDM_ALIAS.ipv4` is listed as `ipv4`.
- Fields that are already included in the query template are shown as grayed out.
- In the Identity and Network templates, `xdm.event.outcome` shows as grayed out. In these templates, the ACTION STATUS and CONNECTION STATUS fields are linked to the `xdm.event.outcome` enum. Therefore, you can't duplicate this field in a query.

6. Click TIME and select a time frame for the query.

7. Click Run to start the query, or click Schedule to run the query at a specific time.

You can also click Continue in XQL to open the XQL Query Builder showing the defined XQL fields. In XQL you have the flexibility to add additional stages and functions that are not available in the Query Builder templates.

8. Review the Results.

The search is limited to 1,000 results. In the Fields column, you can see all of the fields that were included in the query in the following order: (1) `_time`, (2) the filtering fields that you defined, and (3) the fields from the template specific fieldset.

**NOTE:**

This order might change if you include a filtering field that is listed in the fieldset. In that case, the field is taken out of the fieldset and ordered at the top of the list with the other filtering fields.

The query is also saved in the Query Center. In the Query Center, you can identify your query by filtering the Created By column and looking in the Query Description column. Queries created from a template are prefixed with the template name.

Example 103. Example

- The following query searches for instances of IP 3.3.3.3 with a source host name equal to host1 or host2. IP is an alias field; therefore, the query searches all fields associated with the alias.

```
IP ADDRESS = 3.3.3.3, SOURCE.HOST.OS = host1|host2
```

- The following query searches for the event outcome success with an event duration value that is not equal to null:

```
EVENT.OUTCOME = XDM_CONST.OUTCOME_SUCCESS, EVENT.DURATION != Empty values
```



#### What to do next

- To edit or rerun the query, click Back to edit to review the template, or Continue in XQL to review the XQL.
- Practice running queries with Query Builder template examples.

#### 6.2.3.4 | Run a free text query

##### Abstract

Query your datasets for free-text strings with the Free text search template.

You can use the Free text template to query your datasets for free-text strings without building a Cortex Query Language (XQL) query. The template queries all of the datasets that are stored in your tenant and returns up to 1,000 results.

##### NOTE:

Free-text search is also available in XQL queries. You can use the **search** stage to query free-text strings in specific datasets, or all of the datasets in your tenant.

How to run a free text query

1. Select Investigation & Response → Search → Query Builder.
2. Under General Search, select Free text.
3. In the Text Contains field, type one or more strings. Separate multiple strings with pipes, which applies the OR operator.
4. Click TIME and select a time frame for the query.

##### NOTE:

Free text search is limited to the last 90 days of data. Specifying a time frame outside of this limitation will cause the query to fail.

5. Click Run to start the query, or click Schedule to run the query at a specific time.

Free text search searches the relevant columns in each dataset. Relevant columns are subject to a change and can vary between datasets.

You can also click Continue in XQL to translate the query with the fields that you specified into XQL. In XQL you have the flexibility to add additional stages and functions that are not available in the Query Builder templates.

6. Review the results.

The searched string is highlighted in the results.

In the Fields column, you can see all of the fields in which the string was discovered. Fields are listed in the following order: (1) `_time`, (2) `_dataset`, and (3) the fields in which the string was discovered, ordered by highest to lowest number of hits.

In the `RAW_DATA` column, click Show more to see the specific row in the dataset in which the string was discovered.

#### What to do next

- To edit or rerun the query, click Back to edit to review the template in the Query Builder, or Continue in XQL to review the XQL.
- Practice running queries with Query Builder template examples.

#### 6.2.3.5 | Query Builder template examples

##### Abstract

To help you feel confident with Query Builder templates, follow our step-by-step examples and tailor them for your environment.

The following examples can help familiarize you with running queries.

Use the Identity template to search for information about a specific user

**Goal:** Search for information about users working on the system.

This example uses the Identity template, but you can apply it to any of the templates. In the example, we run multiple queries that narrow down our search results and find the required information we require.

Query 1: Search for information about all users

1. Select Investigation & Response → Search → Query Builder.



2. Select the Identity template.
3. Specify USER = \* and do not select Empty values.

This searches for all users, and excludes empty values or strings from the results. The **USER** field is an alias so all associated fields are also searched.

4. Specify TIME â Last 7D.
5. Click Run.

In the Results page, scroll through the table to find a value or string that you want to investigate further. If you are not receiving results, you can broaden the TIME to Last 30D.

In this example, the results returned information about USER66 in the XDM.SOURCE.USER.USERNAME column. To refine the search for information about this user, run another query.

#### Query 2: Search for information about a specific user

1. Copy the term that you want to search, in this case USER66.
  2. Click Back to edit.
- The Identity template opens with the original search options.
3. Click Add field and select SOURCE.USER.USERNAME.
  4. Specify SOURCE.USER.USERNAME = USER66 and do not select Empty values.
  5. Click Run.

The Results page provides more information about USER66.

Look through the results for anything you would like to investigate further. In this example, there is information about the operations performed by this user in the XDM.EVENT.OPERATION column. We can refine the search to see all FILE\_REMOVE operations for USER66.

#### Query 3: Search for FILE\_REMOVE operations for a specific user

1. Click Back to edit.
2. Click Add field and select EVENT.OPERATION.
3. Specify EVENT.OPERATION = and select XDM\_CONST.OPERATION\_TYPE\_FILE\_REMOVE from the list.
4. Click Run.

Review the Results page and continue to refine your search by using this method.

#### Use the Network template to search for hosts triggering threat events in the United States

**Goal:** Search for information about source hosts in the United States that caused threat events over the last 7 days.

#### Query 1: Search for network information in the United States

1. Select Investigation & Response â Search â Query Builder.
  2. Select the Network template.
  3. Specify COUNTRY = United States and do not select Empty values.
- This searches for network activity in the United States, and excludes empty values or strings from the results.
4. Specify TIME â Last 7D.
  5. Click Run.

In the Results page, scroll through the table to find a value or string for which you would like to find more information.

In this example, the results returned information about XDM.EVENT.TYPE = threat for host DC3ENX4FGC07 in the XDM.SOURCE.HOST.HOSTNAME column. To refine the search, run another query.

#### Query 2: Search for information about a specific host and event type

1. Copy the term that you want to search, in this case DC3ENX4FGC07.
  2. Click Back to edit.
- The Network template opens with the original search options.
3. Click Add field and select EVENT.TYPE.



4. Specify EVENT.TYPE = threat and do not select Empty values.
5. Click Add field and select SOURCE.HOST.HOSTNAME.
6. Specify SOURCE.HOST.HOSTNAME = DC3ENX4FGC07 and do not select Empty values.
7. Click Run.

The Results page provides more information about EVENT.TYPE = threat actions from host DC3ENX4FGC07.

To investigate further we could run another query, or in this case, investigate the causality chain of the event. In the search results, right-click and Investigate Causality Chain.

Use the Free text template to search for an IP address

**Goal:** Search for information about IP address 175.18.7.29 in the last 24 hours.

1. Select Investigation & Response → Search → Query Builder.
2. Select the Free text template.
3. Specify Text Contains = 175.18.7.29.
4. Specify TIME → Last 24H.
5. Click Run.

In the Results page the searched string is highlighted. In the Fields column, you can see all of the fields in which the string was discovered. In the RAW\_DATA column, click Show more to see the specific row in the dataset in which the string was discovered.

If you want to deepen your search you can Continue in XQL, which opens an XQL search with the fields you defined in the template. You can add stages and functions to the XQL that narrow down your search.

#### 6.2.4 | Overview of the Query Center

Abstract

View information about the In Progress and Completed queries that were run on the tenant.

The Query Center displays information about all queries that were run on the tenant, and the queries that are currently In Progress. The Query Center displays the following tabs:

- **Query History**

View and manage all completed Cortex Query Language (XQL) and Graph Search queries. On this tab you can view query results, re-run and adjust queries, and schedule when a query runs. You can also see details of cancelled queries, including the query type and source, and the name of the user who cancelled the query.

- **Active Queries**

View and manage all queries that are currently In Progress on the tenant. You can view details about a running query, including the user who ran the query, the context from which it ran, the source of the query, and the amount of time that the query has been running. From this tab you can also cancel active queries.

**NOTE:**

- Very short queries might not be listed.
- You cannot cancel correlation queries.
- The default retention period for historic queries is aligned with issue retention.

#### 6.2.4.1 | Edit and run queries in Query Center

Abstract

Learn more about viewing the results of a query, modifying a query, and rerunning queries from Query Center.

From the Query Center you can take action on the Completed and In Progress queries that are running on your tenant.

Right-click a query to see the available options, where some of the options differ depending on the type of query you've selected. The pivot (right-click) options described below are some of the ones that may require further explanation.

**NOTE:**

If query limits are applied to your tenant, the number of concurrent running queries is limited per user. If query usage is reaching the defined limit, a system message warns you that a high query load is impacting performance. If you exceed the limit, new queries are blocked until query usage drops. You can view all active queries under Query Center → Active Queries, and cancel queries to reduce the load.



#### [View the results of a query](#)

You can view the original results of an XQL query when it was originally run in the Query Builder and added to the Query Center.

1. Select Investigation & Response → Search → Query Center → Query History.

2. Identify the XQL query by looking in the Query Name and Query Description columns.

The Query Description column displays the parameters that were defined for a query. If necessary, use the filter on the column to reduce the number of queries displayed.

Queries that were created from a Query Builder template are prefixed with the template name.

3. Right-click anywhere in the XQL query row and select Show results.

You have the option to Show results in new tab or Show results in same tab.

4. (Optional) Export to file to export the results to a tab-separated values (TSV) file.

5. (Optional) Perform additional investigation on the issues.

Right-click a value in the results table to see the options for further investigation.

#### [Run a query](#)

You can run a query for a Graph Search query.

1. Select Investigation & Response → Search → Query Center → Query History.

2. Identify the Graph Search query by looking in the Query Name and Query Description columns.

The Query Description column displays the parameters that were defined for a query. If necessary, use the filter on the column to reduce the number of queries displayed.

3. Right-click anywhere in the Graph Search query row and select Run query.

You have the option to Run in same tab or Show in new tab.

4. (Optional) The Graph Search results are displayed in a graph format by default. You can toggle to Table to view the results in a table format. In addition, you can always export the graph results using the icon at the top of the page to a PNG, SVG, or TSV file. Table results can only be exported to a TSV file.

5. (Optional) Perform additional investigation on the graph or table results.

On the graph results, you can either hover or select different nodes for further investigation. While in the table results, you can select any cell in the table for further investigation.

#### [Modify a query](#)

After you view the query results of an XQL query or run a Graph Search query as explained in the tasks above, you can change your search parameters to refine the search results or correct a search parameter.

- For queries created in XQL, type your changes in the XQL query field where the original query is listed and the results are displayed in the Query Results tab. After modifying the query, you can run, schedule, or save the query.
- For queries created with a Query Builder template, the defined parameters are shown at the top of the Results page. Select Back to edit to modify the query with the template format or Continue in XQL to open the query in XQL.
- For Graph Search queries, the graph results are displayed. Click anywhere in the Graph Search query interface, where your existing query is defined, to display the complete query, update your query, and rerun the search.

#### [Schedule a query to run](#)

You can schedule an XQL query to run on or before a specific date. Cortex AgentIX creates a new query in the Query Center, and when the query completes, it displays a notification in the notification bar.

How to schedule a query

1. Select Investigation & Response → Search → Query Center → Query History.

2. Right-click anywhere in the query and then select Schedule.

3. Choose a schedule option and the date and time that the query should run:

- Run one time query on a specific date
- Run query by date and time: Schedule a recurring query.

4. Click OK to schedule the query.



Cortex AgentiX creates a new query and schedules it to run on or by the selected date and time.

##### 5. View the status of the scheduled query on the Scheduled Queries page.

You can also make changes to the query, edit the frequency, view when the query will next run, or disable the query. For more information, see [Manage scheduled queries](#).

[Cancel a query](#)

#### **NOTE:**

You can cancel your own queries. To cancel queries run by other users, you must have View/Edit permissions for Configurations → [Query Management](#). By default, Instance administrators have View/Edit permission.

On the Active Queries tab you can cancel one or more In Progress queries. You might want to cancel long-running queries, or cancel queries to reduce tenant consumption. If query limits are applied to your tenant and you exceed the defined limit of concurrent running queries, new queries are blocked until the number of active queries falls below the threshold. Canceling active queries allows you to unblock and run new queries.

How to cancel a query

1. Select Investigation & Response → Search → Query Center → Active Queries.

2. Select one or more queries and click Cancel Selected Queries.

#### **NOTE:**

- Cancelled queries show a Canceled status. You can see details of all canceled queries in the Query History tab.
- You cannot cancel correlation rule queries.
- If you cancel a scheduled query, only the current query is cancelled. Future recurrences of the scheduled query are not affected.

[6.2.4.1.1 | Query Center reference information](#)

Abstract

Descriptions of the fields in the Query Center table.

The table below lists the common fields in the Query Center, where the options differ for an XQL query versus a Graph Search query.

#### **NOTE:**

Certain fields are exposed and hidden by default. An asterisk (\*) is beside every field that is exposed by default.

Query Center table

Field	Description
BQL	Indicates whether the Cortex Query Language (XQL) query was created by the native search.  Native search has been deprecated; this field allows you to view data for XQL queries performed before deprecation.
COMPUTE UNIT USAGE	For XQL queries, indicates the number of query units that were used to execute the API query and Cold Storage query.
ISSUED BY *	For XQL queries, indicates the user who ran or scheduled the query. For Graph Search queries, indicates the user who ran the query.
DURATION (SEC)	Number of seconds it took to execute the XQL query.
EXECUTION ID	Unique identifier of XQL and Graph Search queries in the tenant. The identifier ID generated for queries executed in Cortex AgentiX and XQL query API.
NUM OF RESULTS*	Number of results returned by the query.



Field	Description
PUBLIC API	Whether the source executing the XQL query was an XQL query API.
QUERY DESCRIPTION*	Query parameters used to run the query.
QUERY ID	Unique identifier of the query.
QUERY NAME*	<ul style="list-style-type: none"> <li>• For saved queries, the Query Name identifies the query specified according to a randomly generated number.             <ul style="list-style-type: none"> <li>◦ XQL queries use the format XQL-QUERY-&lt;number&gt;, such as XQL-QUERY-12.</li> <li>◦ Graph Search queries use the format Graph-Query-&lt;number&gt;, such as Graph-Query-1247.</li> </ul> </li> <li>• For scheduled queries, the Query Name identifies the auto-generated name of the parent XQL query. Scheduled queries also display an icon to the left of the name to indicate that the XQL query is recurring.</li> </ul> <div data-bbox="977 843 1264 921" style="border: 1px solid #ccc; padding: 5px; display: inline-block;">  Query - 2432   </div>
QUERY STATUS*	<p>Status of the query, where the options differ based on the query type:</p> <ul style="list-style-type: none"> <li>• XQL queries:             <ul style="list-style-type: none"> <li>◦ Queued: The query is queued and will run when there is an available slot.</li> <li>◦ Running</li> <li>◦ Failed</li> <li>◦ Partially completed: The query was stopped after exceeding the maximum number of permitted results. The default results for a Cortex Data Model (XDM) query or an XQL dataset query is limited to 1000, when no limit is explicitly stated in the query. This applies to basic queries with no stages except the <b>fields</b> stage. This default limit does not apply to widgets, Correlation Rules, public APIs, saved queries, or scheduled queries, where the limit is a maximum of 1,000,000 results. Queries based on legacy templates are limited to 10,000 results. To reduce the number of results returned, you can adjust the query settings and rerun.</li> <li>◦ Stopped: The query was stopped by an administrator.</li> <li>◦ Completed</li> <li>◦ Deleted: The query was pruned.</li> </ul> </li> <li>• Graph Search queries:             <ul style="list-style-type: none"> <li>◦ Failed</li> <li>◦ Completed</li> </ul> </li> </ul>
QUERY SYNTAX	The exact syntax used to write the query.
RESULTS SAVED*	For XQL queries, you can choose whether to save the query results, so the output of the field is either Yes or No. Yet, for Graph Search queries, the results can't be saved and must be run each time again, so the field is always No.



Field	Description
SIMULATED COMPUTE UNITS	Number of XQL query units that were used to execute the Hot Storage query.
Source	Source from which the query was run, for example Playbook, Report, or Investigation.
Source ID	ID of the source from where the query was run.
Source Name	Name of the source from where the query was run.
TIMESTAMP*	Date and time the query was created.
XQL	Indicates whether the XQL query was created by an XQL search.

### 6.2.5 | Manage scheduled queries

#### Abstract

Learn how to manage your scheduled and recurring queries.

The Scheduled Queries page displays information about your scheduled and recurring queries. From this page, you can edit scheduled query parameters, view previous executions, disable, and remove scheduled queries. Right-click a query to see the available options.

#### View executed queries

1. Select Investigation & Response → Search → Scheduled Queries.
  2. Locate the scheduled query for which you want to view previous executions.
- If necessary, use the Filter to reduce the number of queries returned.
3. Right-click anywhere in the query row, and select Show executed queries.

Cortex AgentX filters the queries on the Query Center.

#### Edit the query frequency

1. Select Investigation & Response → Search → Scheduled Queries.
  2. Locate the scheduled query that you want to edit.
- If necessary, use the Filter to reduce the number of queries returned.
3. Right-click anywhere in the query row and then select Edit.
  4. Adjust the schedule settings, and then click OK.

#### 6.2.5.1 | Scheduled Queries reference information

#### Abstract

Descriptions of the fields in the Scheduled Queries table.

The table below lists the common fields in the Scheduled Queries page.

#### NOTE:

Certain fields are exposed and hidden by default. An asterisk (\*) is beside every field that is exposed by default.

Scheduled Queries table



Field	Description
BQL	<p>Whether the query was created by the native search.</p> <p>Native search has been deprecated, this field allows you to view data for queries performed before deprecation.</p>
ISSUED BY	User who ran or scheduled the query.
MITRE ATT&CK TACTIC	MITRE ATT&CK tactics tagged in the scheduled query.
MITRE ATT&CK TECHNIQUE	MITRE ATT&CK techniques tagged in the scheduled query.
NEXT EXECUTION	<ul style="list-style-type: none"> <li>For queries that are scheduled to run at a specific frequency, this displays the next execution time.</li> <li>For queries that were scheduled to run at a specific time and date, this field will show <b>None</b>.</li> </ul>
PUBLIC API	Whether the source executing the query was an XQL query API.
QUERY DESCRIPTION	Query parameters used to run the query.
QUERY ID	Unique identifier of the query.
QUERY NAME	<ul style="list-style-type: none"> <li>For saved queries, the Query Name identifies the query specified by the administrator.</li> <li>For scheduled queries, the Query Name identifies the auto-generated name of the parent query. Scheduled queries also display an icon to the left of the name to indicate that the query is recurring.</li> </ul> <div style="text-align: right; padding-right: 20px;">  <b>Query - 2432</b> </div>
QUERY SYNTAX	The exact syntax used to write the query.
SCHEDULE TIME	Frequency or time at which the query was scheduled to run.
XQL	Whether the query was created by XQL search.

### 6.2.6 | Manage your personal query library

#### Abstract

Cortex AgentIX provides as part of the Query Library a personal library for saving and managing your own queries.

Cortex AgentIX provides as part of the Query Library a personal query library for saving and managing your own queries. When creating a query in XQL Search or managing your queries from the Query Center, you can save queries to your personal library. You can also decide whether the query is shared with others (on the same tenant) in their Query Library or unshare it, so it is only visible to you. You can also view the queries that are shared by others (on the same tenant) in your Query Library.

The queries listed in your Query Library have different icons to help you identify the different states of the queries:



-  Created by me and unshared.
-  Create by me and shared.
-  Created by someone else and shared.

The Query Library contains a powerful search mechanism that enables you to search in any field related to the query, such as the query name, description, creator, query text, and labels. In addition, adding a label to your query enables you to search for these queries using these labels in the Query Library.

How to add a query to your personal query library

1. Save a query to your personal query library.

You can do this in two ways:

- **From the Query Builder**

1. Select Investigation & Response → Search → Query Builder → XQL.
2. In the XQL query field, define the parameters of your query.
3. Select Save as → Query to Library.

- **From the Query Center**

1. Select Investigation & Response → Search → Query Center.
2. Locate the query that you want to save to your personal query library.
3. Right-click anywhere in the query row, and select Save query to library.

2. Set these parameters.

- **Query Name:** Specify a unique name for the query. Query names must be unique in both private and shared lists, which includes other people's queries.
- **Query Description (Optional):** Specify a descriptive name for your query.
- **Labels (Optional):** Specify a label that is associated with your query. You can select a label from the list of predefined labels or add your label and then select Create Label. Adding a label to your query enables you to search for queries using this label in the Query Library.
- **Share with others:** You can either set the query to be private and only accessible by you (default) or move the toggle to Share with others the query, so that other users using the same tenant can access the query in their Query Library.

3. Click Save.

A notification appears confirming that the query was saved successfully to the library, and closes on its own after a few seconds.

The query that you added is now listed as the first entry in the Query Library. The query editor is opened to the right of the query.

4. Other available options.

As needed, you can return to your queries in the Query Library to manage your queries. Here are the actions available to you.

- Edit the name, description, labels, and parameters of your query by selecting the query from the Query Library, hovering over the line in the query editor that you want to edit, and selecting the edit icon to edit the text.
- **Search query data and metadata:** Use the Query Library's powerful search mechanism that enables you to search in any field related to the query, such as the query name, description, creator, query text, and label. The Search query data and metadata field is available at the top of your list of queries in the Query Library.
- **Show:** Filter the list of queries from the Show menu. You can filter by the Palo Alto Networks queries provided with Cortex AgentIX, filter by the queries Created by Me, or filter by the queries Created by Others. To view the entire list, Select all (default).
- **Save as new:** Duplicate the query and save it as a new query. This action is available from the query menu by selecting the 3 vertical dots.
- **Share with others:** If your query is currently unshared, you can share with other users on the same tenant your query, which will be available in their Query Library. This action is only available from the query menu by selecting the 3 vertical dots when your query is unshared.
- **Unshare:** If your query is currently shared with other users, you can Unshare the query and remove it from their Query Library. This action is only available from the query menu by selecting the 3 vertical dots when your query is shared with others. You can only Unshare a query that you created. If another user created the query, this option is disabled in the query menu.
- **Delete the query:** You can only delete queries that you created. If another user created the query, this option is disabled in the query menu when selecting the 3 vertical dots.



## 6.2.7 | Legacy Query Builder

### Abstract

Learn more about the entities in the Legacy Query Builder.

#### NOTE:

We recommend using the Query Builder in New mode to take advantage of the Query Builder templates and the ability to search the full Cortex Data Model (XDM).

In Legacy mode, the Query Builder searches predefined datasets only. To search the full XDM, switch to New mode or select XQL Search.

The Legacy Query Builder provides queries for the following types of entities:

- Process: Search on process execution and injection by process name, hash, path, command line arguments, and more. See Create process query.
- File: Search on file creation and modification activity by file name and path. See Create file query.
- Network: Search network activity by IP address, port, host name, protocol, and more. See Create network query.
- Image Load: Search on module load into process events by module IDs and more. See Create image load query.
- Registry: Search on registry creation and modification activity by key, key value, path, and data. See Create registry query.
- Event Log: Search Windows event logs and Linux system authentication logs by username, log event ID (Windows only), log level, and message. See Create event log query.
- Network Connections: Search security event logs by firewall logs, endpoint raw data over your network. See Create network connections query.
- Authentications: Search on authentication events by identity, target outcome, and more. See Create authentication query.
- All Actions: Search across all network, registry, file, and process activity by endpoint or process. See Query across all entities.

The Query Builder also provides flexibility for both on-demand query generation and scheduled queries.

### 6.2.7.1 | Create authentication query

#### Abstract

Learn more about creating a query to investigate any authentication activity.

From the Query Builder, you can investigate authentication activity across all ingested authentication logs and data.

Some examples of authentication queries you can run include:

- Authentication logs by severity
- Authentication logs by the event message
- Authentication logs for a specific source IP address

How to build an authentication query

1. From Cortex AgentiX , select Investigation & Response → Search → Query Builder.

2. Select AUTHENTICATION.

3. Enter the search criteria for the authentication query.

By default, Cortex AgentiX will return the activity that matches all the criteria you specify. To exclude a value, toggle the = option to !=.

4. Choose when to run the query.

Select the calendar icon to schedule a query to run on or before a specific date or Run to run the query immediately and view the results in the Query Center.

While the query is running, you can always navigate away from the page and a notification is sent when the query completes. You can also Cancel the query or run a new query, where you have the option to Run only new query (cancel previous) or Run both queries.

5. When you are ready, view the results of the query. For more information, see Review XQL query results.

### 6.2.7.2 | Create event log query

#### Abstract

Learn more about creating a query to investigate Windows and Linux event log attributes and investigate event logs across endpoints.

From the Query Builder you can search Windows and Linux event log attributes and investigate event logs across endpoints with a Cortex XDR agent installed.



Some examples of event log queries you can run include:

- Critical level messages on specific endpoints.
- Message descriptions with specific keywords on specific endpoints.

How to build an event log query

1. From Cortex AgentiX , select Investigation & Response → Search → Query Builder.
2. Select EVENT LOG.
3. Enter the search criteria for your Windows or Linux event log query.

Define any event attributes for which you want to search. By default, Cortex XDR will return the events that match the attribute you specify. To exclude an attribute value, toggle the = option to !=. Attributes are:

- PROVIDER NAME: The provider of the event log.
- USERNAME: The username associated with the event.
- EVENT ID: The unique ID of the event.
- LEVEL: The event severity level.
- MESSAGE: The description of the event.

To specify an additional exception (match this value except), click the + to the right of the value and specify the exception value.

4. (*Optional*) Limit the scope to an endpoint or endpoint attributes:

Specify one or more of the following attributes: Use a pipe (|) to separate multiple values.

Use an asterisk (\*) to match any string of characters.

- HOST: HOST NAME, HOST IP address, HOST OS, HOST MAC ADDRESS, or INSTALLATION TYPE.
- INSTALLATION TYPE can be either Cortex XDR agent or Data Collector.
- PROCESS: NAME, PATH, CMD, MD5, SHA256, USER NAME, SIGNATURE, or PID.

5. Specify the time period for which you want to search for events.

Options are Last 24H (hours), Last 7D (days), Last 1M (month), or select a Custom time period.

6. Choose when to run the query.

Select the calendar icon to schedule a query to run on or before a specific date or Run to run the query immediately and view the results in the Query Center.

While the query is running, you can always navigate away from the page, and a notification is sent when the query completes. You can also Cancel the query or run a new query, where you have the option to Run only new query (cancel previous) or Run both queries.

7. When you are ready, view the results of the query. For more information, see Review XQL query results.

#### 6.2.7.3 | Create file query

Abstract

Learn more about creating a query to investigate the connections between file activity and endpoints.

From the Query Builder you can investigate connections between file activity and endpoints. The Query Builder searches your logs and endpoint data for the file activity that you specify. To search for files on endpoints instead of file-related activity, build an XQL query. For more information, see How to build XQL queries.

Some examples of file queries you can run include:

- Files modified on specific endpoints.
- Files related to process activity that exist on specific endpoints.

How to build a file query

1. From Cortex AgentiX , select Investigation & Response → Search → Query Builder.
2. Select FILE.
3. Enter the search criteria for the file events query.



- File activity: Select the type or types of file activity you want to search: All, Create, Read, Rename, Delete, or Write.
- File attributes: Define any additional process attributes for which you want to search. Use a pipe (|) to separate multiple values (for example `notepad.exe|chrome.exe`). By default, Cortex AgentX will return the events that match the attribute you specify. To exclude an attribute value, toggle the = option to !=. Attributes are:
  - NAME: File name.
  - PATH: Path of the file.
  - PREVIOUS NAME: Previous name of a file.
  - PREVIOUS PATH: Previous path of the file.
  - MD5: MD5 hash value of the file.
  - SHA256: SHA256 hash value of the file.
  - ACTION\_DISK\_DRIVER\_NAME: The driver where the file was created.
  - FILE\_SYSTEM\_TYPE: Operating system type where the file was run.
  - ACTION\_IS\_VFS: Denotes if the file is on a virtual file system on the disk. This is relevant only for files that are written to disk.
  - DEVICE TYPE: Type of device used to run the file: Unknown, Fixed, Removable Media, CD-ROM.
  - DEVICE SERIAL NUMBER: Serial number of the device type used to run the file.

To specify an additional exception (match this value except), click the + to the right of the value and specify the exception value.

#### 4. (Optional) Limit the scope to a specific acting process:

Select +PROCESS and specify one or more of the following attributes for the acting (parent) process.

Use a pipe (|) to separate multiple values. Use an asterisk (\*) to match any string of characters.

- NAME: Name of the parent process.
- PATH: Path to the parent process.
- CMD: Command-line used to initiate the process, including any arguments, up to 128 characters.
- MD5: MD5 hash value of the process.
- SHA256: SHA256 hash value of the process.
- USER NAME: User who executed the process.
- SIGNATURE: Signing status of the parent process: Signature Unavailable, Signed, Invalid Signature, Unsigned, Revoked, Signature Fail.
- SIGNER: Entity that signed the certificate of the parent process.
- PID: Process ID of the parent process.
- Run search for process, Causality, and OS actors—also referred to as the causality group owner (CGO)—is the parent process in the execution chain that the Cortex XDR agent identified as being responsible for initiating the process tree. The OS actor is the parent process that creates an OS process on behalf of a different indicator. By default, this option is enabled to apply the same search criteria to initiating processes. To configure different attributes for the parent or initiate the process, clear this option.

#### 5. (Optional) Limit the scope to an endpoint or endpoint attributes:

Select +Host and specify one or more of the following attributes:

- HOST: HOST NAME, HOST IP address, HOST OS, HOST MAC ADDRESS, or INSTALLATION TYPE.  
INSTALLATION TYPE can be either Cortex XDR agent or Data Collector.
- PROCESS: NAME, PATH, CMD, MD5, SHA256, USER NAME, SIGNATURE, or PID.

Use a pipe (|) to separate multiple values. Use an asterisk (\*) to match any string of characters.

#### 6. Specify the time period for which you want to search for events.

Options are Last 24H (hours), Last 7D (days), Last 1M (month), or select a Custom time period.

#### 7. Choose when to run the query.

Select the calendar icon to schedule a query to run on or before a specific date or Run to run the query immediately and view the results in the Query Center.



While the query is running, you can always navigate away from the page and a notification is sent when the query completes. You can also Cancel the query or run a new query, where you have the option to Run only new query (cancel previous) or Run both queries.

8. When you are ready, view the results of the query. For more information, see Review XQL query results.

#### 6.2.7.4 | Create image load query

##### Abstract

Learn more about create a query to investigate the connections between image load activity, acting processes, and endpoints.

From the Query Builder, you can investigate connections between image load activity, acting processes, and endpoints.

Some examples of image load queries you can run include:

- Module load into process events by module path or hash.

##### How to build an image load query

1. From Cortex AgentiX , select Investigation & Response â Search â Query Builder.

2. Select IMAGE LOAD.

3. Enter the search criteria for the image load activity query.

- Type of image activity: All, Image Load, or Change Page Protection.
- Identifying information about the image module: Full Module Path, Module MD5, or Module SHA256.

By default, Cortex AgentiX will return the activity that matches all the criteria you specify. To exclude a value, toggle the = option to !=.

4. (Optional) To limit the scope to a specific source, click the + to the right of the value and specify the exception value.

Specify one or more attributes for the source.

Use a pipe (|) to separate multiple values. Use an asterisk (\*) to match any string of characters.

- NAME: Name of the parent process.
- PATH: Path to the parent process.
- CMD: Command-line used to initiate the process, including any arguments, up to 128 characters.
- MD5: MD5 hash value of the process.
- SHA256: SHA256 hash value of the process.
- USER NAME: User who executed the process.
- SIGNATURE: Signing status of the parent process: Signature Unavailable, Signed, Invalid Signature, Unsigned, Revoked, Signature Fail.
- SIGNER: Entity that signed the certificate of the parent process.
- PID: Process ID of the parent process.

Run search for both the process and the Causality actor: The causality actorâ€ also referred to as the causality group owner (CGO)â€ is the parent process in the execution chain that the app identified as being responsible for initiating the process tree. Select this option if you want to apply the same search criteria to the causality actor. If you clear this option, you can then configure different attributes for the causality actor.

5. (Optional) Limit the scope to an endpoint or endpoint attributes:

Specify one or more of the following attributes: Use a pipe (|) to separate multiple values.

Use an asterisk (\*) to match any string of characters.

- HOST: HOST NAME, HOST IP address, HOST OS, HOST MAC ADDRESS, or INSTALLATION TYPE.  
INSTALLATION TYPE can be either Cortex XDR agent or Data Collector.
- PROCESS: NAME, PATH, CMD, MD5, SHA256, USER NAME, SIGNATURE, or PID.

6. Specify the time period for which you want to search for events.

Options are Last 24H (hours), Last 7D (days), Last 1M (month), or select a Custom time period.

7. Choose when to run the query.



Select the calendar icon to schedule a query to run on or before a specific date or Run to run the query immediately and view the results in the Query Center.

While the query is running, you can always navigate away from the page and a notification is sent when the query completes. You can also Cancel the query or run a new query, where you have the option to Run only new query (cancel previous) or Run both queries.

8. When you are ready, view the results of the query. For more information, see Review XQL query results.

#### 6.2.7.5 | Create network connections query

##### Abstract

Learn more about creating a query to investigate the connections between firewall logs, endpoints, and network activity.

From the Query Builder, you can investigate network events stitched across endpoints and the Palo Alto Networks Next-Generation Firewall logs.

Some examples of a network query you can run include:

- Source and destination of a process.
- Network connections that included a specific App ID
- Processes that created network connections.
- Network connections between specific endpoints.

##### How to build a network connection query

1. From Cortex AgentiX , select Investigation & Response → Search → Query Builder.
2. Select NETWORK CONNECTIONS.
3. Enter the search criteria for the network events query.

- Network attributes: Define any additional process attributes for which you want to search. Use a pipe (|) to separate multiple values (for example 80 | 8080). By default, Cortex AgentiX will return the events that match the attribute you specify. To exclude an attribute value, toggle the = option to !=. Options are:
  - APP ID: App ID of the network.
  - PROTOCOL: Network transport protocol over which the traffic was sent.
  - SESSION STATUS
  - FW DEVICE NAME: Firewall device name.
  - FW RULE: Firewall rule.
  - FW SERIAL ID: Firewall serial ID.
  - PRODUCT
  - VENDOR

To specify an additional exception (match this value except), click the + to the right of the value and specify the exception value.

4. (Optional) To limit the scope to a specific source, click the + to the right of the value and specify the exception value.

Specify one or more attributes for the source.

Use a pipe (|) to separate multiple values. Use an asterisk (\*) to match any string of characters.



- HOST NAME: Name of the source.
- HOST IP: IP address of the source.
- HOST OS: Operating system of the source.
- PROCESS NAME: Name of the process.
- PROCESS PATH: Path to the process.
- CMD: Command-line used to initiate the process, including any arguments, up to 128 characters.
- MD5: MD5 hash value of the process.
- SHA256: SHA256 hash value of the process.
- PROCESS USER NAME: User who executed the process.
- SIGNATURE: Signing status of the parent process: Signature Unavailable, Signed, Invalid Signature, Unsigned, Revoked, Signature Fail.
- PID: Process ID of the parent process.
- IP: IP address of the process.
- PORT: Port number of the process.
- USER ID: ID of the user who executed the process.
- Run search for both the process and the Causality actor: The causality actor is also referred to as the causality group owner (CGO) is the parent process in the execution chain that the app identified as being responsible for initiating the process tree. Select this option if you want to apply the same search criteria to the causality actor. If you clear this option, you can then configure different attributes for the causality actor.

#### 5. (Optional) Limit the scope to a destination.

Use a pipe (|) to separate multiple values. Use an asterisk (\*) to match any string of characters.

Specify one or more of the following attributes:

- REMOTE IP: IP address of the destination.
- COUNTRY: Country of the destination.
- Destination TARGET HOST,NAME, PORT, HOST NAME, PROCESS USER NAME, HOST IP, CMD, HOST OS, MD5, PROCESS PATH, USER ID, SHA256, SIGNATURE, or PID

#### 6. Specify the time period for which you want to search for events.

Options are Last 24H (hours), Last 7D (days), Last 1M (month), or select a Custom time period.

#### 7. Choose when to run the query.

Select the calendar icon to schedule a query to run on or before a specific date or Run to run the query immediately and view the results in the Query Center.

While the query is running, you can always navigate away from the page and a notification is sent when the query completes. You can also Cancel the query or run a new query, where you have the option to Run only new query (cancel previous) or Run both queries.

#### 8. When you are ready, view the results of the query. For more information, see Review XQL query results.

### 6.2.7.6 | Create network query

#### Abstract

Learn more about creating a query to investigate the connections between network activity, acting processes, and endpoints.

From the Query Builder, you can investigate connections between network activity, acting processes, and endpoints.

Some examples of a network query you can run include:

- Network connections to or from a specific IP address and port number.
- Processes that created network connections.
- Network connections between specific endpoints.

#### How to build a network query

1. From Cortex AgentIX , select Investigation & Response â Search â Query Builder.



2. Select NETWORK.

3. Enter the search criteria for the network events query.

- Network traffic type: Select the type or types of network traffic issues you want to search: Incoming, Outgoing, or Failed.
- Network attributes: Define any additional process attributes for which you want to search. Use a pipe (|) to separate multiple values (for example 80 | 8080). By default, Cortex AgentX will return the events that match the attribute you specify. To exclude an attribute value, toggle the = option to !=. Options are:
  - REMOTE COUNTRY: Country from which the remote IP address originated.
  - REMOTE IP: Remote IP address related to the communication.

**NOTE:**

When you run the query, depending on the outcome of the results, the value specified in this field might be displayed in the dst\_ip field in the query results. This occurs if an RDP event is recorded whereby a user connected from the source IP to the destination IP.

- REMOTE PORT: Remote port used to make the connection.
- LOCAL IP: Local IP address related to the communication. Matches can return additional data if a machine has more than one NIC.
- LOCAL PORT: Local port used to make the connection.
- PROTOCOL: Network transport protocol over which the traffic was sent.

To specify an additional exception (match this value except), click the + to the right of the value and specify the exception value.

4. (Optional) To limit the scope to a specific source, click the + to the right of the value and specify the exception value.

Specify one or more attributes for the source.

Use a pipe (|) to separate multiple values. Use an asterisk (\*) to match any string of characters.

- NAME: Name of the parent process.
- PATH: Path to the parent process.
- CMD: Command-line used to initiate the process, including any arguments, up to 128 characters.
- MD5: MD5 hash value of the process.
- SHA256: SHA256 hash value of the process.
- USER NAME: User who executed the process.
- SIGNATURE: Signing status of the parent process: Signature Unavailable, Signed, Invalid Signature, Unsigned, Revoked, Signature Fail.
- SIGNER: Entity that signed the certificate of the parent process.
- PID: Process ID of the parent process.
- Run search for process, Causality, and OS actors: The causality actor—also referred to as the causality group owner (CGO)—is the parent process in the execution chain that the Cortex XDR agent identified as being responsible for initiating the process tree. The OS actor is the parent process that creates an OS process on behalf of a different indicator. By default, this option is enabled to apply the same search criteria to initiating processes. To configure different attributes for the parent or initiate the process, clear this option.

5. (Optional) Limit the scope to an endpoint or endpoint attributes:

Specify one or more of the following attributes: Use a pipe (|) to separate multiple values.

Use an asterisk (\*) to match any string of characters.

- HOST: HOST NAME, HOST IP address, HOST OS, HOST MAC ADDRESS, or INSTALLATION TYPE.
- INSTALLATION TYPE can be either Cortex XDR agent or Data Collector.
- PROCESS: NAME, PATH, CMD, MD5, SHA256, USER NAME, SIGNATURE, or PID.

6. Specify the time period for which you want to search for events.

Options are Last 24H (hours), Last 7D (days), Last 1M (month), or select a Custom time period.

7. Choose when to run the query.

Select the calendar icon to schedule a query to run on or before a specific date or Run to run the query immediately and view the results in the Query Center.

While the query is running, you can always navigate away from the page and a notification is sent when the query completes. You can also Cancel the query or run a new query, where you have the option to Run only new query (cancel previous) or Run both queries.



8. When you are ready, view the results of the query. For more information, see Review XQL query results.

#### 6.2.7.7 | Create process query

##### Abstract

Learn more about creating a query to investigate connections between processes, child processes, and endpoints.

From the Query Builder you can investigate connections between processes, child processes, and endpoints.

For example, you can create a process query to search for processes executed on a specific endpoint.

##### How to build a process query

1. From Cortex AgentiX , select Investigation & Response → Search → Query Builder.

2. Select PROCESS.

3. Enter the search criteria for the process query.

- Process action: Select the type of process action you want to search: On process Execution or Injection into another process.
- Process attributes: Define any additional process attributes for which you want to search.

Use a pipe (|) to separate multiple values. Use an asterisk (\*) to match any string of characters.

By default, Cortex AgentiX will return results that match the attribute you specify. To exclude an attribute value, toggle the operator from = to !=. Attributes are:

- NAME: Name of the process. For example, `notepad.exe`.
- PATH: Path to the process. For example, `C:\windows\system32\notepad.exe`.
- CMD: Command-line used to initiate the process including any arguments, up to 128 characters.
- MD5: MD5 hash value of the process.
- SHA256: SHA256 hash value of the process.
- USER NAME: User who executed the process.
- SIGNATURE: Signing status of the process: Signature Unavailable, Signed, Invalid Signature, Unsigned, Revoked, Signature Fail.
- SIGNER: Signer of the process.
- PID: Process ID.
- PROCESS\_FILE\_INFO: Metadata of the process file, including file property details, file entropy, company name, encryption status, and version number.
- PROCESS\_SCHEDULED\_TASK\_NAME: Name of the task scheduled by the process to run in the Task Scheduler.
- PROCESS\_TOKEN\_INFORMATION: Bitwise token of the process privileges.
- DEVICE TYPE: Type of device used to run the process: Unknown, Fixed, Removable Media, CD-ROM.
- DEVICE SERIAL NUMBER: Serial number of the device type used to run the process.

To specify an additional exception (match this value except), click the + to the right of the value and specify the exception value.

4. (Optional) Limit the scope to a specific acting process:

Select +PROCESS and specify one or more of the following attributes for the acting (parent) process.



- NAME: Name of the parent process.
- PATH: Path to the parent process.
- CMD: Command-line used to initiate the parent process including any arguments, up to 128 characters.
- MD5: MD5 hash value of the parent process.
- SHA256: SHA256 hash value of the process.
- USER NAME: User who executed the process.
- SIGNATURE: Signing status of the parent process: Signed, Unsigned, N/A, Invalid Signature, Weak Hash
- SIGNER: Entity that signed the certificate of the parent process.
- PID: Process ID of the parent process.
- Run search on process, Causality and OS actors: The causality actor—also referred to as the causality group owner (CGO)—is the parent process in the execution chain that the Cortex XDR agent identified as being responsible for initiating the process tree. The OS actor is the parent process that creates an OS process on behalf of a different initiator. By default, this option is enabled to apply the same search criteria to initiating processes. To configure different attributes for the parent or initiate a process,

5. (Optional) Limit the scope to an endpoint or endpoint attributes:

Select +HOST and specify one or more of the following attributes:

- HOST: HOST NAME, HOST IP address, HOST OS, HOST MAC ADDRESS, or INSTALLATION TYPE.  
INSTALLATION TYPE can be Cortex XDR agent.
- PROCESS: NAME, PATH, CMD, MD5, SHA256, USER NAME, SIGNATURE, or PID.

6. Specify the time period for which you want to search for events.

Options are Last 24H (hours), Last 7D (days), Last 1M (month), or select a Custom time period.

7. Choose when to run the query.

Select the calendar icon to schedule a query to run on or before a specific date or Run to run the query immediately and view the results in the Query Center.

While the query is running, you can always navigate away from the page and a notification is sent when the query completes. You can also Cancel the query or run a new query, where you have the option to Run only new query (cancel previous) or Run both queries.

8. When you are ready, view the results of the query. For more information, see Review XQL query results.

#### 6.2.7.8 | Create registry query

##### Abstract

Learn more about creating a query to investigate connections between registry activity, processes, and endpoints.

From the Query Builder you can investigate connections between registry activity, processes, and endpoints.

Some examples of a registry query you can run include:

- Modified registry keys on specific endpoints.
- Registry keys related to process activity that exist on specific endpoints.

##### How to build a registry query

1. From Cortex AgentiX , select Investigation & Response → Search → Query Builder.
2. Select REGISTRY.
3. Enter the search criteria for the registry events query.



- Registry action: Select the type or types of registry actions you want to search: Key Create, Key Delete, Key Rename, Value Set, or Value Delete.
- Registry attributes: Define any additional registry attributes for which you want to search. By default, Cortex AgentX will return the events that match the attribute you specify. To exclude an attribute value, toggle the = option to !=. Attributes are:
  - KEY NAME: Registry key name.

**IMPORTANT:**

Ensure the KEY NAME is entered as a real registry key name, and not as a symbolic link. Otherwise, the query will not retrieve results.

Example 104.

Instead of `HKEY_LOCAL_MACHINE\System\CurrentControlSet`, which is a symbolic link, use `HKEY_LOCAL_MACHINE\System\ControlSet001`.

Example 105.

Instead of `HKEY_CURRENT_USER`, use `HKEY_USERS\<SID>`, where SID is either a SID of the current user or an asterisk (\*) to represent any SID.

- DATA: Registry key data value.
- KEY PREVIOUS NAME: Name of the registry key before modification.
- VALUE NAME: Registry value name.

To specify an additional exception (match this value except), click the + to the right of the value and specify the exception value.

4. (Optional) To limit the scope to a specific source, click the + to the right of the value and specify the exception value.

Specify one or more attributes for the source.

Use a pipe (|) to separate multiple values. Use an asterisk (\*) to match any string of characters.

- NAME: Name of the parent process.
- PATH: Path to the parent process.
- CMD: Command-line used to initiate the process including any arguments, up to 128 characters.
- MD5: MD5 hash value of the process.
- SHA256: SHA256 hash value of the process.
- USER NAME: User who executed the process.
- SIGNATURE: Signing status of the parent process: Signature Unavailable, Signed, Invalid Signature, Unsigned, Revoked, Signature Fail.
- SIGNER: Entity that signed the certificate of the parent process.
- PID: Process ID of the parent process.
- Run search for process, Causality, and OS actors: The causality actor—also referred to as the causality group owner (CGO)—is the parent process in the execution chain that the Cortex XDR agent identified as being responsible for initiating the process tree. The OS actor is the parent process that creates an OS process on behalf of a different indicator. By default, this option is enabled to apply the same search criteria to initiating processes. To configure different attributes for the parent or initiate the process, clear this option.

5. (Optional) Limit the scope to an endpoint or endpoint attributes:

Specify one or more of the following attributes: Use a pipe (|) to separate multiple values.

Use an asterisk (\*) to match any string of characters.

- HOST: HOST NAME, HOST IP address, HOST OS, HOST MAC ADDRESS, or INSTALLATION TYPE.
- INSTALLATION TYPE can be either Cortex XDR agent or Data Collector.
- PROCESS: NAME, PATH, CMD, MD5, SHA256, USER NAME, SIGNATURE, or PID.

6. Specify the time period for which you want to search for events.

Options are Last 24H (hours), Last 7D (days), Last 1M (month), or select a Custom time period.

7. Choose when to run the query.

Select the calendar icon to schedule a query to run on or before a specific date or Run to run the query immediately and view the results in the Query Center.



While the query is running, you can always navigate away from the page and a notification is sent when the query completes. You can also Cancel the query or run a new query, where you have the option to Run only new query (cancel previous) or Run both queries.

8. When you are ready, view the results of the query. For more information, see Review XQL query results.

#### 6.2.7.9 | Query across all entities

##### Abstract

From the Cortex AgentX management console, you can search for endpoints and processes across all endpoint activity.

From the Query Builder you can perform a simple search for hosts and processes across all file events, network events, registry events, process events, event logs for Windows, and system authentication logs for Linux.

Some examples of queries you can run across all entities include:

- All activities on a host
- All activities initiated by a process on a host

##### How to build a query

1. From Cortex AgentX , select Investigation & Response → Search → Query Builder.

2. Select ALL ACTIONS.

3. (Optional) Limit the scope to a specific acting process:

Select Add Process to your search, and specify one or more of the following attributes for the acting (parent) process. Use a pipe (|) to separate multiple values. Use an asterisk (\*) to match any string of characters.

Field	Description
NAME	Name of the parent process.
PATH	Path to the parent process.
CMD	Command line used to initiate the parent process including any arguments, up to 128 characters.
MD5	MD5 hash value of the parent process.
SHA256	SHA256 hash value of the process.
USER NAME	User who executed the process.
SIGNATURE	Signing status of the parent process: Signed, Unsigned, N/A, Invalid Signature, Weak Hash.
SIGNER	Entity that signed the certificate of the parent process.
PID	Process ID of the parent process.
Run search on process, Causality and OS actors	The causality actor, also referred to as the causality group owner (CGO), is the parent process in the execution chain that the agent identified as being responsible for initiating the process tree. The OS actor is the parent process that creates an OS process on behalf of a different initiator. By default, this option is enabled to apply the same search criteria to initiating processes. To configure different attributes for the parent or initiating process, clear this option.

4. (Optional) Limit the scope to an endpoint or endpoint attributes:



Select Add Host to your search and specify one or more of the following attributes:

- HOST: HOST NAME, HOST IP address, HOST OS, HOST ADDRESS, or INSTALLATION TYPE.
- INSTALLATION TYPE can be either an agent, or data collector.
- PROCESS: NAME , PATH , CMD , MD5 , SHA256 , USER NAME , SIGNATURE, or PID.

Use a pipe (|) to separate multiple values. Use an asterisk (\*) to match any string of characters.

#### 5. Specify the time period for which you want to search for events.

Options are Last 24H (hours), Last7D (days), Last1M (month), or select a Custom time period.

#### 6. Choose when to run the query.

Select the calendar icon to schedule a query to run on or before a specific date or Run the query immediately and view the results in the Query Center.

While the query is running, you can always navigate away from the page and a notification is sent when the query completes. You can also Cancel the query or run a new query, where you have the option to Run only new query (cancel previous) or Run both queries.

#### 7. When ready, view the results in a query.

## 6.3 | Stages

### Abstract

Learn more about the Cortex Query Language supported stages.

Stages perform certain operations in evaluating queries. For example, the **dataset** stage specifies a dataset to run the query. Commonly used stages include **dataset**, **fields**, **filters**, **join**, and **sort**. The stages supported in Cortex Query Language are detailed below.

### 6.3.1 | alter

#### Abstract

Learn more about the Cortex Query Language **alter** stage.

Review the following topic:

- Understanding string manipulation in XQL

#### Syntax

```
alter <field1> = <function value1> [, <field2> = <function value2>, ...]
```

#### Description

The **alter** stage is used to change the values of an existing field (column) or to create a new field (column) based on constant values or existing fields (columns). The **alter** stage does this by assigning a value to a field name based on the returned value of the specified function. The field does not have to be known to the dataset or preset schema that you are querying. Further, you can overwrite the current value for a known field using this stage.

After defining a field using the **alter** stage, you can apply other stages, such as filtering, to the new field or field value.

#### Examples

Given three username fields, use the coalesce function to return a username value in the **default\_username** field, making sure to never have a **default\_username** that is **root**.

```
dataset = xdr_data
| fields actor_primary_username,
  os_actor_primary_username,
  causality_actor_primary_username
| alter default_username = coalesce(actor_primary_username,
  os_actor_primary_username,
  causality_actor_primary_username)
| filter default_username != "root"
```

### 6.3.2 | arrayexpand

#### Abstract

Learn more about the Cortex Query Language **arrayexpand** stage.



## Syntax

```
arrayexpand <array_field> [limit <limit number>]
```

## Description

The `arrayexpand` stage expands the values of a multi-value array field into separate events and creates one record in the result set for each item in the array, up to a `<limit number>` of records.

## Example

Suppose you have a dataset with a single row like this:

Uid	Username	Array_values
123456	ajohnson	[1,2,3,4,5,6,7,8,9,0]

Then if you run an `arrayexpand` stage using the `array_values` field, with a limit of 3, the result set includes the following records:

```
dataset=my_dataset
| arrayexpand array_values limit 3
```

Uid	Username	Array_values
123456	ajohnson	2
123456	ajohnson	1
123456	ajohnson	3

## NOTE:

The result records created by `arrayexpand` are in no particular order. However, you can use the `sort` stage to sort the results:

```
dataset=my_dataset
| arrayexpand array_values
| sort asc array_values
```

## 6.3.3 | bin

### Abstract

Learn more about the Cortex Query Language `bin` stage to group events by quantity or time span.

## Syntax

- `Quantity`

```
bin <field> bins = <number>

• Time Span

bin <field> span = <time> [timeshift = <epoch time> [timezone = "<time zone>"]]
```

## Description

The `bin` stage enables you to group events by quantity or time span. The most common use case is for timecharts.

You can add the `bin` stage to your queries using two different formats depending on whether you are grouping events by quantity or time span. Currently, the `bin` stage is only supported using the equal sign (=) operator in your queries without any boolean operators (`and`, `or`).

When you group events of a particular field by quantity, the `bin` stage is used with `bins` to define how to divide the events.

When you group events of a particular field by time, the `bin` stage is used with `span = <time>`, where `<time>` is a combination of a number and time suffix. Set one time suffix from the list of available options listed in the table below. In addition, you can define a particular start time for grouping the events in your query according to the Unix epoch time by setting `timeshift = <epoch time>` `timezone = "<time zone>"`, which are both optional. You can



configure the `<time zone>` offset using an hours offset, such as `+08:00`, or using a time zone name from the List of Supported Time Zones, such as "America/Chicago". The query still runs without defining the epoch time or time zone. If no `timeshift = <epoch time>` `timezone = "<time zone>"` is set, the query runs according to last time set in the log.

#### NOTE:

When you group events by quantity, the `<field>` in the `bin` stage must be a number, and when you group by time, the `<field>` must be a date type. Otherwise, your query will fail.

Time Suffixes

Time Suffix	Description
MS	milliseconds
S	seconds
M	minutes
H	hours
D	days
W	weeks
MO	months
Y	years

#### NOTE:

The time suffix is not case sensitive.

Examples

- Quantity Example

Return a maximum of 1,000 `xdr_data` records with the events of the `action_total_upload` field grouped by 50MB. Records with the `action_total_upload` value set to 0 or null are not included in the results.

```
dataset = xdr_data
| filter action_total_upload != 0 and action_total_upload != null
| bin action_total_upload bins = 50
| limit 1000
```

- Time Span Examples

- With a time zone configured using an hours offset:

Return a maximum of 1,000 `xdr_data` records with the events of the `_time` field grouped by 1-hour increments starting from the epoch time `1615353499`, and includes a time zone using an hours offset of `+08:00`.

```
dataset = xdr_data
| bin _time span = 1h timeshift = 1615353499 timezone = +08:00
| limit 1000
```

- With a time zone name configured:

Return a maximum of 1,000 `xdr_data` records with the events of the `_time` field grouped by 1-hour increments starting from the epoch time `1615353499`, and includes an "America/Los\_Angeles" time zone.

```
dataset = xdr_data
| bin _time span = 1h timeshift = 1615353499 timezone = America/Los_Angeles
| limit 1000
```



### 6.3.4 | call

#### Abstract

Learn more about the Cortex Query Language **call** stage to reference a predefined query from the Query Library.

#### Syntax

```
call "<name of predefined query>" [<param_name1> = <value1> <param_name2> = <value2>....]
```

#### Description

The **call** stage is used to reference a predefined query from the Query Library, including your Personal Query Library. In addition, if your query includes parameters you can reference them in the **call** stage using the syntax **<param\_name1> = <value1> <param\_name2> = <value2>....**. When using parameters in your **call** stage, you need to ensure that a query already exists that uses these parameters.

#### Example without Parameters

For the predefined query called "CreateRole operation parsed to fields", returns a maximum of 100 records, where the **accessKeyId** equals "1234".

```
call "CreateRole operation parsed to fields"
| filter accessKeyId = "1234"
| limit 100
```

#### Example with Parameters

Using the same example above, this example shows how to use the same **call** stage with parameters. This example assumes that there is a query that is already saved with a parameter **\$key\_id = "1234"**.

Saved query:

```
dataset = dataset_name
| filter field_name = $key_id
```

Query to run with using parameters:

```
call "CreateRole operation parsed to fields" key_id = "1234"
| limit 100
```

### 6.3.5 | comp

#### Abstract

Learn more about the Cortex Query Language **comp** stage that precedes functions calculating statistics.

#### Syntax

```
comp <aggregate function1> (<field>) [as <alias>][,<aggregate function2>(<field>) [as <alias>],....] [by <field1>[,<field2>...]]
[addrawdata = true|false [as <target field>]]
```

#### Description

The **comp** stage precedes functions calculating statistics for results to compute values over a group of rows and return a single result for a group of rows.

- Aggregation functions, such as **sum**, **min**, and **max**
- Approximate aggregate functions, such as **approx\_count** or **approx\_top**

At least one of the comp aggregate functions or comp approximate aggregate functions must be used. Yet, it's also possible to define a comp stage with both types of aggregate functions.

Use approximate aggregate functions to produce approximate results, instead of exact results used with regular aggregate functions, which are more scalable in terms of memory usage and time.

Use the **alias** clause to provide a column label for the **comp** results, and is optional.

The **by** clause identifies the rows in the result set that will be aggregated. This clause is optional. Provide one or more fields to this clause. All fields with matching values are used to perform the aggregation. For example, if you had records such as:

```
number,id,product
100,"se1","A55"
50,"se1","A60"
50,"se1","A60"
25,"se2","A55"
25,"se2","A60"
```

The you can aggregate on the number column, and perform aggregation based on matching values in the id and/or product column. So if you sum the number column by the id column, you would get two results:



- 200 for "se1"
- 50 for "se2"

If you summed by id and product, you would get:

- 100 for "se1" and "A55" (there are no matching pairs).
- 100 for "se1" and "A60" (there is one matching pair).
- 25 for "se2" and "A55" (there are no matching pairs).
- 25 for "se2" and "A60" (there are no matching pairs).

In addition, you can configure whether the raw data events are displayed by setting `addrawdata` to either `true` or `false` (default), which are used to configure the final `comp` results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

#### Wildcard Aggregates

You can use a wildcard to perform an aggregate for every field contained in the result set, except for the field(s) specified in the `by` clause.

##### **NOTE:**

Wildcards are only supported with aggregate functions and not approximate aggregate functions.

The syntax for this is:

```
comp <aggregate function>(*) as * [by [asc|desc] <field1>[,<field2>...]]  
[addrawdata = true|false as <target field>]
```

For wildcards to work, all of the fields contained in the result set that are not identified in the `by` clause must be aggregatable.

#### Examples

Sum the `action_total_download` values for all records with matching pairs of values for the `actor_process_image_path` and `actor_process_command_line` fields. The query calculates a maximum of 100 `xdr_data` records and includes a `raw_data` column listing the raw data events used to display the final `comp` results.

```
dataset = xrdr_data  
| fields actor_process_image_path as Process_Path,  
actor_process_command_line as Process_CMD,  
action_total_download as Download  
| filter Download > 0  
| limit 100  
| comp sum(Download) as total by Process_Path, Process_CMD addrawdata = true as raw_data
```

Using the `panw_ngfw_traffic_raw` dataset, sum the `bytes_total`, `bytes_received`, and `bytes_sent` values for every record contained in the result set with a matching value for `source_ip`. The query calculates a maximum of 1000 `xdr_data` records and includes a `raw_data` column listing the raw data events used to display the final `comp` results.

##### **NOTE:**

The `comp` stage runs on 1000 raw data events, but only a 100 will be displayed in the `raw_data` column.

```
dataset = panw_ngfw_traffic_raw  
| fields bytes_total, bytes_received, bytes_sent, source_ip  
| limit 1000  
| comp sum(*) as * by source_ip addrawdata = true as raw_data
```

#### comp Aggregate Functions

The aggregate functions you can use with the `comp` stage are:



- avg
- count
- count\_distinct
- earliest
- first
- last
- latest
- list
- max
- median
- min
- stddev\_population
- stddev\_sample
- sum
- values
- var

#### **comp Approximate Aggregate Functions**

The approximate aggregate functions you can use with the **comp** stage are:

- approx\_count
- approx\_quantiles
- approx\_top

### 6.3.6 | **config**

#### Abstract

Learn more about the Cortex Query Language **config** stage that configures the query behavior.

#### Syntax

```
config <function>
```

#### Description

The **config** stage configures the query behavior. It must be used with one of the config Functions. This stage must be presented as the first stage in the query.

#### config Functions

These functions you can use with the **config** stage:

- case\_sensitive
- timeframe
- max\_runtime\_minutes

#### 6.3.6.1 | **case\_sensitive**

#### Abstract

Learn more about the Cortex Query Language **case\_sensitive** config stage.

#### Syntax

```
config case_sensitive = true | false
```



## Description

The `case_sensitive` configuration identifies whether field values are evaluated as case sensitive or case insensitive. The `config case_sensitive` stage must be added at the beginning of the query. You can also add another `config case_sensitive` stage when adding a join or union stage to a query.

If you do not provide this stage in your query, the default behavior is `false`, and case is not considered when evaluating field values.

Things to keep in mind about before implementing this stage

- The Settings → Configurations → XQL Configuration → Case Sensitivity (case\_sensitive) setting can overwrite this `case_sensitive` configuration for all fields in the application, except for BIOCs, which will remain case insensitive no matter what this setting is set to.
- The `config case_sensitive` stage can't be used to compare a field to an inner query. In this situation, ensure to use the `lowercase` or `uppercase` functions on the field and inner query stages and functions syntax.

Example 106.

This query won't provide the correct results of comparing the `agent_hostname` field with the inner query:

```
config case_sensitive = false
| dataset = xdr_data
| fields agent_hostname
| filter agent_hostname in (dataset = <lookup dataset> | fields agent_hostname)
```

This query will provide the correct output:

```
config case_sensitive = false
| dataset = xdr_data
| fields agent_hostname
| filter lowercase(agent_hostname) in (dataset = <lookup dataset> | alter lower_agent_hostname = lowercase(agent_hostname) | fields lower_agent_hostname)
```

- The `config case_sensitive` stage can't be used to compare a field to an array that contains non-literal strings, for example a field name or function.

Example 107.

The results of this example are true, where the left side (`uppercase("a")`) is lowercase as it's not an array, and the right side ((`"x"`, `"A"`)) is also an array that contains only literal strings.

```
| alter field_name = if(uppercase("a") in ("x", "A"), true, false)
```

Example 108.

The results of this example are false, where the left side (`uppercase("a")`) is lowercase as it's not an array, and the right side (`"x"`, `uppercase("a")`) is an array that contains a function (`uppercase("a")`).

```
| alter field_name = if(uppercase("a") in ("x", uppercase("a")), true, false)
```

## Examples

```
config case_sensitive = true
| dataset = xdr_data
| fields actor_process_image_name as apin
| filter apin != NULL and apin contains "python"
| limit 100
```

### 6.3.6.2 | timeframe

#### Abstract

Cortex Query Language `timeframe` configuration enables performing searches within a specific time frame from the query execution.

#### Syntax

- Exact Time

```
config timeframe between "<Year-Month-Day H:M:S Â±Timezone>" and "<Year-Month-Day H:M:S Â±Timezone>"
```

- Relative Time

```
config timeframe = <number><time unit>
config timeframe between "<+|-><number><time unit>" and "now"
config timeframe between "begin" and "<+|-><number><time unit>"
config timeframe between "<+|-><number><time unit>" and "<+|-><number><time unit>"
```



## Description

The `timeframe` configuration enables you to perform searches within a specific time frame from the query execution. The results for the time frame are based on times listed in the `_Time` column in the results table.

You can add the `timeframe` configuration to your queries using different formats depending on whether the time frame you are setting is an exact time or relative time.

When you set an exact time, include the `config timeframe` details: between "`<Year-Month-Day H:M:S Â±Timezone>`" and "`<Year-Month-Day H:M:S Â±Timezone>`". The `Â±Timezone` format is: `Â±xxxx`. When you do not configure a timezone, the default is `UTC`. The exact time is based on a static time frame according to when the query is sent.

When you set a relative time, you have a few options for setting the `config timeframe`, where the syntax `<+|->` indicates whether to go back (-) or forward (+) in time. The default is back (-).

- `<number><time unit>`

Enables setting a static time frame according to when the query is sent, where you choose the `<time unit>` from the available time unit options listed in the table below.

- `between "<+|-><number><time unit>" and "now"`

Enables setting a time frame between a defined start time, where you choose the `<time unit>` from the available time unit options listed in the table below, and the end time as the time the query is run with the preset keyword "now".

- `between "begin" and "<+|-><number><time unit>"`

Enables setting a time frame between a preset start time according to the Unix epoch time 00:00:00 UTC on 1 January 1970 with the "begin" keyword, and a defined ending time, where you choose the `<time unit>` from the available time unit options listed in the table below.

- `between "<+|-><number><time unit>" and "<+|-><number><time unit>"`

Enables setting a time frame between a defined starting and ending time, where you choose the `<time unit>` from the available time unit options listed in the table below.

## IMPORTANT:

When a query includes any inner queries, the inner queries receives its time frame from the outer query unless the inner query has a separate time frame defined.

## Connection to the time period in the Query Builder

When using the Query Builder to define a query, the time period can be set at the top right of the query window using the time picker, and the default is 24 hours. Whenever the time period is changed in the query window, the `config timeframe` is automatically set to the time period defined, but this won't be visible as part of the query. Only if you manually type in the `config timeframe` will this be seen in the query.

## Available time units

Time Unit	Description
S	seconds
M	minutes
H	hours
D	days
W	weeks
MO	months
Y	years



## NOTE:

The time unit is not case sensitive.

### Examples

#### Relative Time

- Example of <number><time unit>

For the last 10 hours from when the query is sent, return a maximum of 100 `xdr_data` records.

```
config timeframe = 10h
| dataset = xdr_data
| limit 100
```

- Example of `between "<+|-><number><time unit>" and "now"`

Since the last two days until now when the query is run, return a maximum of 100 `xdr_data` records.

```
config timeframe between "2d" and "now"
| dataset = xdr_data
| limit 100
```

- Example of `between "begin" and "<+|-><number><time unit>"`

Since the Unix epoch time 00:00:00 UTC on 1 January 1970 until the past 2 years when the query is run, return a maximum of 100 `xdr_data` records.

```
config timeframe between "begin" and "2y"
| dataset = xdr_data
| limit 100
```

- Example of `between "<+|-><number><time unit>" and "<+|-><number><time unit>"`

Since the last four days until the next 5 days when the query is run, return a maximum of 100 `xdr_data` records.

```
config timeframe between "-4d" and "+5d"
| dataset = xdr_data
| limit 100
```

### Exact Time

From April 1, 2021 at 9:00 a.m. UTC -02:00 until April 2, 2021 at 10:00 a.m. UTC -02:00, return a maximum of 100 `xdr_data` records.

```
config timeframe between "2021-04-01 09:00:00 -0200" and "2021-04-02 10:00:00 -0200"
| dataset = xdr_data
| limit 100
```

### 6.3.6.3 | `max_runtime_minutes`

#### Abstract

Learn more about the Cortex Query Language `max_runtime_minutes` config stage.

#### Syntax

```
config max_runtime_minutes = <number> | ...
```

#### Description

The `max_runtime_minutes` function is used to override the default query duration timeout (60 minutes). You can increase the query duration timeout up-to the administrator defined value, as defined in Settings → Configurations → General → Query Management → Query Limits.

Only integer values are supported for this field. In addition, the query timeout is an approximate value.

For more information about query limits, see XQL query management.

### Examples

```
config max_runtime_minutes = 90
| dataset = xdr_data
| fields actor_process_image_name as apin
| filter apin != NULL and apin contains "python"
| limit 100
```

### 6.3.7 | `dedup`

#### Abstract



Learn more about the Cortex Query Language **dedup** stage that removes duplicate occurrences of field values.

#### Syntax

```
dedup <field1>[,<field2>, ...] by asc | desc <field>
```

#### Description

The **dedup** stage removes all records that contain duplicate values (or duplicate sets of values) from the result set. The record that is returned is identified by the **by** clause, which selects the record by either the first or last occurrence of the field specified in this clause.

#### NOTE:

The dedup stage can only be used with fields that contain numbers or strings.

#### Examples

Return unique values for the **actor\_primary\_username** field. For any given field value, return the first chronologically occurring record.

```
dataset = xdr_data
| fields actor_primary_username as apu
| filter apu != null
| dedup apu by asc _time
```

Return the last chronologically occurring record for any given **actor\_primary\_username** value.

```
dataset = xdr_data
| fields actor_primary_username as apu
| filter apu != null
| dedup apu by desc _time
```

Return the first occurrence seen by for any given **actor\_primary\_username**. field value.

```
dataset = xdr_data
| fields actor_primary_username as apu
| filter apu != null
| dedup apu by asc apu
```

Return unique groups of **actor\_primary\_username** and **os\_actor\_primary\_username** field values. For each unique grouping, return the pair that first appears on a record with a non-NULL **action\_file\_size** field.

```
dataset = xdr_data
| fields actor_primary_username as apu,
  os_actor_primary_username as oapu,
  action_file_size as afs
| filter apu != null and afs != null
| dedup apu, oapu by asc afs
```

### 6.3.8 | fields

#### Abstract

Learn more about the Cortex Query Language **fields** stage that defines the fields returned in the result set.

#### Syntax

- Dataset Queries

```
fields [-] <field_1> [as <name1>], <field_2> [as <name2>], ...
```

- Cortex Data Model (XDM) Queries

```
  • fields [-] <field_1> [as <name1>], <field_2> [as <name2>], ...
```

```
  • fields [-] fieldset.xdm_<fieldset name1>, fieldset.xdm_<fieldset name2>, ...
```

```
  • Combination of both options above are supported with fields and fieldsets in any order:
```

```
    fields [-] fieldset.xdm_<fieldset name1> , <field_1> [as <name1>], fieldset.xdm_<fieldset name2>network , <field_2>
```

#### NOTE:

When creating XDM queries, the raw dataset fields are accessible by **<dataset>.<field>**, such as **fields amazon\_eks\_raw.logStream**.

#### Description

The **fields** stage declares which fields are returned in the result set, including name changes. If this stage is used, then subsequent stages can operate only on the fields identified by this stage. The syntax for this stage differs depending on the type of query you are running.



## Both Dataset and XDM Queries

For both dataset and XDM queries, your `fields` stage syntax can include the following elements:

### Wildcards

Use a wildcard (\*) to include all fields that match the pattern, where wildcards can only be added at the beginning or end of a string. The following table explains the different scenarios for using wildcards in fields with examples:

#### NOTE:

Wildcards are not supported in fieldsets.

Wildcard Scenarios	Examples
Adding at the end of a field.	<ul style="list-style-type: none"><li>• Dataset Queries<ul style="list-style-type: none"><li>◦ <code>event_*</code></li></ul></li><li>• XDM Queries<ul style="list-style-type: none"><li>◦ <code>xdm.source*</code></li><li>◦ <code>xdm.source.*</code></li><li>◦ <code>xdm.source.a*</code></li></ul></li></ul>
Adding at the beginning of a field, when there is no period anywhere else in the field.	<ul style="list-style-type: none"><li>• Supported syntax<ul style="list-style-type: none"><li>◦ <code>*ipv4</code></li></ul></li><li>• Unsupported syntax<ul style="list-style-type: none"><li>◦ <code>*.ipv4</code></li><li>◦ <code>*source.ipv4</code></li></ul></li></ul>
Adding at both the beginning and end of a field has the same limitations as using it at the beginning of a field.	<ul style="list-style-type: none"><li>• Supported syntax<ul style="list-style-type: none"><li>◦ <code>*ipv4*</code></li></ul></li><li>• Unsupported syntax<ul style="list-style-type: none"><li>◦ <code>*.ipv4*</code></li><li>◦ <code>*ipv4.*</code></li><li>◦ <code>*source.ipv4*</code></li></ul></li></ul>
Workaround syntax using the ` character can be used, which does not support the auto-suggest feature during XQL query creation.	<ul style="list-style-type: none"><li>• <code>*`ipv4`</code></li><li>• <code>*`source.ipv4`</code></li></ul>

### Minus Character

Use a minus character (-) to exclude a field from the result set. For example, `| fields - <field1>, <field2>` will exclude both `<field1>` and `<field2>` fields in your query results.

The following system fields cannot be excluded and are always displayed, if they exist:

- Dataset queries: `_time`, `_insert_time`, `_raw_log`, `_product`, `_vendor`, `_tag`, `_snapshot_id`, `_snapshot_log_count`, `_snapshot_collection_ts`, `_id`
- XDM queries: `_time`

### As Clause

Use the `as` clause to set an alias for a field. If you use the `as` clause, then subsequent stages must use that alias to refer to the field.



## XDM Queries

For XDM queries, your `fields` stage syntax can include the following additional elements:

### Fieldsets

Use a `fieldset` within the `fields` stage to refine queries on the XDM by limiting the analysis to a specific set of fields. Fieldsets contain a group of related fields, for example, the `fieldset.xdm_endpoint` includes fields that are related to endpoints.

The `xdm_core` fieldset contains fields typically queried by users, including commonly used event, source, and target fields. When no specific fields are specified in a query, the following fields will be returned by default: `_time`, `xdm.event.type`, `xdm.event.description`, `xdm.event.operation`, `xdm.event.operation_sub_type`, `xdm.event.outcome`, `xdm.source.host.hostname`, `xdm.source.user.username`, `xdm.source.user.user_type`, `xdm.source.sent_bytes`, `xdm.source.agent.identifier`, `xdm.source.user_agent`, `xdm.source.process.name`, `xdm.source.process.executable.path`, `xdm.source.process.executable.filename`, `xdm.source.ipv4`, `xdm.source.port`, `xdm.target.host.hostname`, `xdm.target.user.username`, `xdm.target.process.executable.path`, `xdm.target.ipv4`, `xdm.target.port`, `xdm.target.user.user_type`, `xdm.target.sent_bytes`, `xdm.target.agent.identifier`, `xdm.target.url`, `xdm.target.domain`, `xdm.target.process.name`, `xdm.target.process.executable.filename`, `xdm.event.outcome_reason`, `xdm.observer.product`, `xdm.event.is_completed`, `xdm.event.duration`

For more information on these fields, see the Cortex Data Model Schema Guide.

### Wildcards

When combining the results of a dataset and XDM query using the join stage, the wildcard (\*) relates to both. For example, this query will return both `datamodel` fields that contain "host" and `xdr_data` fields that contain "host".

```
datamodel
| join (dataset=xdr_data) as x xdm.original_event_id = x.event_id
| fields *host*
```

### Dataset Query Example

Return the `action_country` field from all `xdr_data` records where the `action_country` field is both not null and not "-". Also include all fields with names that match `event_*` except for `event_type`.

```
dataset = xdr_data
| fields action_country as ac
| fields event_*
| fields - event_type
| filter ac != null and ac != "-"
```

### XDM Query Example

Return the XDM fields that are related to the network (`fieldset.xdm_network`), fields that are related to endpoints (`fieldset.xdm_endpoint`), and the `xdm.alert.name` field.

```
datamodel dataset = xdr_data
| fields fieldset.xdm_network, fieldset.xdm_endpoint, xdm.alert.name
```

### XDM Query using a Wildcard

Return the XDM fields that are related to the `xdm.source.*` and `xdm.email.*` fields, where the `xdm.source.user.username` is `newman`.

```
datamodel dataset = xdr_data
| filter xdm.source.user.username = "newman"
| fields xdm.source.*, xdm.email.*
```

## 6.3.9 | filter

### Abstract

Learn more about the Cortex Query Language `filter` stage that narrows down the displayed results.

### Syntax

```
filter <boolean expr>
```

### Description

The `filter` stage identifies which data records should be returned by the query. Filters are boolean expressions that can use a wide range of functions and operators to express the filter. If a record matches the filter as the filter expression returns `true` when applied to the record, the record is returned in the query's result set.

The functions you can use with a filter are described in Functions. For a list of supported operators, see Supported operators.



## Single vs triple double quotes behavior

Cortex AgentiX enables you to use single double quotes ("<text>") or triple double quotes ("""<text>""") when defining your XQL syntax for string manipulation. This specific syntax is used with different stages, functions, and operators, with or without wildcards. Typically, the **alter** and **filter** stages are used with single or triple double quotes.

### Using single double quotes

Single double quotes ("<text>") include the following functionality:

- Treats the string value literally.
- Wildcards using the asterisk (\*) are processed as XQL wildcards, and match any sequence of characters.
- Escape sequences, such as \n (new line) or \t (tab), are not processed and are treated as plain characters.

Example 109.

"\test\" means to look for \test\

### Using triple double quotes

Triple double quotes ("""<text>""") include the following functionality:

- Enables regex-style pattern matching and escape sequence interpretation.
- Escape sequences, such as \n (new line) or \t (tab), are processed.
- Wildcards using the asterisk (\*) are processed as XQL wildcards, and match any sequence of characters.

Example 110.

"""\\test\\"" means to look for \test\

### Understanding the results:

- The double backslashes (\\\) at the beginning becomes a single backslash (\) as it's processed as an escaped backslash.
- **test** is interpreted as literal.
- The double backslashes (\\\) at the end becomes a single backslash (\) as it's processed as an escaped backslash.

### Query example using filter

When using the **filter** stage, you can use both single ("<text>") and triple ("""<text>""") double quotes when specifying string values. The difference lies in how special characters and pattern matching are interpreted.

The examples provided are based on the following data table for a dataset called **test\_dataset**:

_TIME	TEST
Mar 26th 2022 19:26:07	12\t3
May 7th 2023 15:16:00	12 3
Jun 8th 2024 16:56:27	1233
Mar 26th 2024 19:26:07	123
Apr 5th 2024 11:21:02	12\t34563
Apr 9th 2025 13:22:22	1233345
May 9th 2025 13:22:22	12 35897



<b>_TIME</b>	<b>TEST</b>
May 30th 2025 21:45:02	116

Example 111.

```
config timeframe = 10y
| dataset = test_dataset
| filter test = "12\t3*"
| fields test
```

Output results table:

<b>_TIME</b>	<b>TEST</b>
Mar 26th 2022 19:26:07	12\t3
Apr 5th 2024 11:21:02	12\t34563

Explanation of results:

The asterisk (\*) in "12\t3\*" means to process the string field as an XQL wildcard by matching any sequence of characters that begins with 12\t3. In addition, the \t characters are not processed as an escape character, but as plain characters.

Example 112.

```
config timeframe = 10y
| dataset = test_dataset
| filter test = """12\t3*"""
| fields test
```

Output results table:

<b>_TIME</b>	<b>TEST</b>
May 7th 2023 15:16:00	12 3
May 9th 2025 13:22:22	12 35897

Explanation of results:

The \t in """12\t3\*"" is processed as a tab escape character. The asterisk (\*) in """12\t3\*"" means to process the string field as an XQL wildcard by matching any sequence of characters that begins with 12<tab>3.

## Dataset Query Examples

Return `xdr_data` records where the `event_type` is `NETWORK` and the `event_sub_type` is `NETWORK_HTTP_HEADER`.

```
dataset = xdr_data
| filter event_type = NETWORK and event_sub_type = NETWORK_HTTP_HEADER
```

### NOTE:

When adding filters to an XQL query, possible field values for enum fields are available using the auto-complete feature. Yet, the autocomplete can only show enum values that are known to the schema. In some cases, on data import an enum value is included that is not known to the defined schema. In this case, the value will appear in the result set as an unknown value, such as `event_type_unknown_4`. Be aware that even though this value appears in the result set, you cannot create a filter using it. For example, this query will fail, even if you know the value appears in your result set:

```
dataset = xdr_data
| filter event_type = event_type_unknown_4
```

When using fields of type `ENUM`, the following syntax is supported:

### Syntax format A



```
| filter event_type = ENUM.FILE
```

## Syntax format B

```
| filter event_type = FILE
```

### XDM Query Examples

Return the XDM fields that are related to the `xdm.source.*` and `xdm.email.*` fields, where the `xdm.source.user.username` is `newman`.

```
datamodel dataset = xdr_data
| filter xdm.source.user.username = "newman"
| fields xdm.source.*, xdm.email.*
```

### XDM CONSTS (ENUMS)

When using fields of type ENUM, you can map values from a predefined list of ENUMs. For example, the field `xdm.network.ip_protocol` is defined as `Enum.IP_PROTOCOL`, so you can assign it values such as `XDM_CONST.IP_PROTOCOL_TCP`. The full list can be found in the automatically suggested values for the relevant fields. This syntax is not mandatory.

```
datamodel dataset = xdr_data
| filter xdm.network.ip_protocol = XDM_CONST.IP_PROTOCOL_TCP
```

For more information on the XDM CONST fields, see the Cortex Data Model Schema Guide.

### XDM Aliases

The Cortex Data Model (XDM) includes aliases. These are predefined sets of fields that can be used to simplify your filter. When the `XDM_ALIAS` keyword is added while writing a query, a list of available predefined aliases and a tooltip are displayed. The tooltip provides more details about the selected alias. The aliases support these Cortex Query Language (XQL) operators: `comparison`, `string`, and `range`.

For example, when you type this query to search the IPv4 field in the XDM,

```
datamodel dataset = xdr_data
| filter XDM_ALIAS.ipv4 = "10.10.10.10"
```

the tooltip displays the fields that will be searched for the alias `XDM_ALIAS.ipv4`:

```
xdm.network.dhcp.ciaddr, xdm.target.ipv4, xdm.network.dhcp.giaddr, xdm.source.ipv4, xdm.intermediate.ipv4,
xdm.network.dhcp.yiaddr, xdm.network.dhcp.siaddr
```

The query above is the equivalent to the following syntax, which does not contain a predefined alias, and displays the rows that match the alias `XDM_ALIAS.ipv4` equaling "10.10.10.10" at least once in the fields that make up the alias:

```
datamodel dataset = xdr_data
| filter xdm.network.dhcp.ciaddr = "10.10.10.10"
or xdm.target.ipv4 = "10.10.10.10"
or xdm.network.dhcp.giaddr = "10.10.10.10"
or xdm.source.ipv4 = "10.10.10.10"
or xdm.intermediate.ipv4 = "10.10.10.10"
or xdm.network.dhcp.yiaddr = "10.10.10.10"
or xdm.network.dhcp.siaddr = "10.10.10.10"
```

In this example, when you type this query to search the IPv4 field in the XDM,

```
datamodel dataset = xdr_data
| filter XDM_ALIAS.ipv4 != "10.10.10.10"
```

the tooltip displays the fields that will be searched for the alias `XDM_ALIAS.ipv4`:

```
xdm.network.dhcp.ciaddr, xdm.target.ipv4, xdm.network.dhcp.giaddr, xdm.source.ipv4, xdm.intermediate.ipv4,
xdm.network.dhcp.yiaddr, xdm.network.dhcp.siaddr
```

The query above is the equivalent to the following syntax, which does not contain a predefined alias, and does not display any rows that match the alias `XDM_ALIAS.ipv4` equaling "10.10.10.10" at least once in the fields that make up the alias:

```
datamodel dataset = xdr_data
| filter xdm.network.dhcp.ciaddr != "10.10.10.10"
and xdm.target.ipv4 != "10.10.10.10"
and xdm.network.dhcp.giaddr != "10.10.10.10"
and xdm.source.ipv4 != "10.10.10.10"
and xdm.intermediate.ipv4 != "10.10.10.10"
and xdm.network.dhcp.yiaddr != "10.10.10.10"
and xdm.network.dhcp.siaddr != "10.10.10.10"
```

## 6.3.10 | getrole

### Abstract

Learn more about the Cortex Query Language `getrole` stage that enriches events with specific roles associated with usernames or endpoints.



#### LICENSE TYPE:

This stage requires an Identity Threat Module license to view the results.

#### IMPORTANT:

This stage is unsupported in BIOCs and real-time Correlation Rules.

##### Syntax

```
getrole <field> [as <alias>]
```

##### Description

The **getrole** stage enriches events with specific roles associated with usernames or endpoints. The **getrole** stage receives as an input a string field that is either a username in the **NETBIOS\SAM** format, such as **mydomain\myuser**, or the agent ID of a host. The agent ID can be found in the **endpoints** dataset as **endpoint\_id** or in the **xdr\_data** dataset as **agent\_id**.

The roles for this field are displayed in a column called **asset\_roles** in the results table. If there is one or more roles associated with the field, the values are represented as a string array, such as **[ 'ADMIN', 'USER' ]**, and are listed in the **asset\_roles** column. If there are no roles, the resulting column is an empty array.

You can also change the name of the column using **as** in the syntax to define an alias: **getrole <field> as <alias>**.

In addition, it is possible to use the **filter** stage with a new **ROLE** prefix to display the results of a particular role using the syntax:

- To include one specific role:

- **filter <field> = ROLE.<role name>**
  - **filter array\_length(arrayfilter(<field>, "@element" = ROLE.<role name> )) > 0**

- To include more than one specific role:

- **filter <field> in (ROLE.<role name1>, ROLE.<role name2>, ....)**

- To exclude one specific role:

- **filter array\_length(arrayfilter(<field>, "@element" = ROLE.<role name> )) = 0**

- To exclude more than one specific role:

- **filter array\_length(arrayfilter(<field>, "@element" in (ROLE.<role name1>, ROLE.<role name2>, ....))) = 0**

##### Examples

Return a maximum of 100 **xdr\_data** records with the enriched events including specific roles associated with usernames. If there are one or more roles associated with the value of the **user\_id** string field column, the output is displayed in the **asset\_roles** column in the results table. Otherwise, the field is empty.

```
dataset = xdr_data
| limit 100
| getrole user_id
```

Return a maximum of 100 **xdr\_data** records of all the powershell executions made by the **SERVICE\_ACCOUNTS** user role in the organization. The first **filter** stage indicates how to filter for the parent process, which is powershell.exe. The **fields** stage indicates the field columns to include in the results table and which ones are renamed in the table: **action\_process\_image\_name** to **process\_name** and **action\_process\_image\_command\_line** to **process\_cmd**. The **getrole** stage indicates the enriched events to include for the specific roles associated with usernames. If the **ROLE.SERVICE\_ACCOUNTS** role is associated with any values in the **actor\_effective\_username** string field column, the row is displayed in the results table. Otherwise, the entire row is excluded from the results table.

```
dataset = xdr_data
| filter event_type = ENUM.PROCESS and event_sub_type = ENUM.PROCESS_START and lowercase(actor_process_image_name) = "powershell.exe"
| fields action_process_image_name as process_name, action_process_image_command_line as process_cmd, event_id, actor_effective_username
| getrole actor_effective_username as user_roles
| filter user_roles = ROLE.SERVICE_ACCOUNTS
| limit 100
```

## 6.3.11 | iploc

##### Abstract

Learn more about the Cortex Query Language **iploc** stage that associates IPv4 addresses of fields to a list of predefined attributes related to the geolocation.

##### Syntax

```
iploc <field>
```



## Description

The `iploc` stage associates the IPv4 address of any field to a list of predefined attributes related to the geolocation. By default, when using this stage in your queries, the geolocation data is added to the results table in these predefined column names: `LOC ASN ORG`, `LOC ASN IS PROXY`, `LOC ASN`, `LOC CITY`, `LOC CONTINENT`, `LOC COUNTRY`, `LOC LATLON`, `LOC REGION`, and `LOC TIMEZONE`.

### NOTE:

The `loc_latlon` field contains a string that is a combination of two floating numbers representing the latitude and longitude separated by a comma, for example, `32.0695,34.7621`.

The following options are available to you when using this stage in your queries:

- You can specify the geolocation fields that you want added to the results table.
- You can append a suffix to the name of the geolocation field column in the results table.
- You can change the name of the geolocation field column in the results table.
- You can also view the geolocation data on a graph type called map, where the `xaxis` is set to either `loc_country` or `loc_latlon`, and the `yaxis` is a number field.

### NOTE:

- The `iploc` stage can only be used with fields that contain numbers or strings.
- To improve your query performance, we recommend that you filter the data in your query before the `iploc` stage is run. In addition, limiting the number of fields in the results table further improves the performance.

## Examples

Return a maximum of 1000 `xdr_data` records with the specific geolocation data associated with the `action_remote_ip` field, where no record with a null value for `action_remote_ip` is included, and displays the name of the city in a column called `city` and a combination of the latitude and longitude in a column called `loc_latlon` with comma-separated values of latitude and longitude.

```
dataset = xdr_data
| limit 1000
| filter action_remote_ip != null
| iploc action_remote_ip loc_city as city, loc_latlon
```

Return a maximum of 1000 `xdr_data` records with all the available geolocation data with the predefined column names, and add the specified `suffix_remote_id` to each predefined column name, where no record with a null value for `action_remote_ip` is included.

```
dataset = xdr_data
| limit 1000
| filter action_remote_ip != null
| iploc action_remote_ip suffix=_remote_id
```

Return a maximum of 1000 `xdr_data` records with the specific geolocation data associated with the `action_remote_ip` field that includes the name of the country (contained in `loc_country`) in a column called `country`, where no record with a null value for either `country` or `action_remote_ip` is included. The `comp` stage is used to count the number of IP addresses per country. The results are displayed in a graph type of `kind` map, where the x-axis represents the `country` and the y-axis the `action_remote_ip`.

```
dataset = xdr_data
| limit 1000
| iploc action_remote_ip loc_country as country
| filter country != null and action_remote_ip != null
| comp count() as ip_count by country
| view graph type = map xaxis = country yaxis = ip_count
```

## 6.3.12 | join

### Abstract

Learn more about the Cortex Query Language `join` stage that combines the results of two queries into a single result set.

### Syntax

```
join conflict_strategy = both|left|right
  type = inner|left|right
  ((<xql_query>
    as <execution_name>
    <boolean_expr>)
```

## Description

The `join()` stage combines the results of two queries into a single result set. This stage is conceptually identical to a SQL join.



Parameter/Clause	Description
<code>conflict_strategy</code>	<p>Identifies the join conflict strategy when there is a conflict in the column names between the 2 result sets which one should be chosen, either:</p> <ul style="list-style-type: none"> <li>• <code>right</code>: The column from the inner <code>join</code> query is used (default), which implements a right outer join.</li> <li>• <code>left</code>: The column from the original result set in the dataset is used, which implements a left outer join.</li> <li>• <code>both</code>: Both columns are used. The original result set column from the dataset keeps the current name, while the inner <code>join</code> query result set column name includes the following suffix added to the current name <code>_joined_10</code>, such as <code>&lt;original column name&gt;_joined_10</code>, and depending on the number of conflicted fields the suffix increases to <code>_joined_11, _joined_12....</code></li> </ul>
<code>type</code>	<p>Identifies the join type.</p> <ul style="list-style-type: none"> <li>• <code>inner</code> Returns all the records in common between the queries that are being joined. This is the default join type.</li> <li>• <code>right</code> Returns all records from the join result set, plus any records from the parent result set that intersect with the join result set.</li> <li>• <code>left</code> Returns all records from the parent result set, plus any records from the join result set that intersect with the parent result set.</li> </ul>
<xql query>	Provides the XQL query to be joined with the parent query.
as <execution_name>	Provides an alias for the join query's result set. For example, if you specify an execution name of <code>join1</code> , and in the join query you return field <code>agent_id</code> , then you can subsequently refer to that field as <code>join1.agent_id</code> .
<boolean_expr>	Identifies the conditions that must be met in order to place a record in the join result set.

#### NOTE:

This stage does not preserve sort order. If you are combining this stage with a sort stage, specify the `sort` stage after the `join`.

#### Examples

Return `microsoft_windows_raw` records, which are combined with the `xdr_data` records to include a new column called `edr`. For the `event_type` set to `EVENT_LOG`, the `actor_process_image_name` and `event_id` fields are returned from all `xdr_data` records, which are then compared to the fields inside the `microsoft_windows_raw` dataset, where `edr.event_id = edr.event_id`, and the results are added to the new `edr` column.

```
dataset = microsoft_windows_raw
| join (dataset = xdr_data | filter event_type = EVENT_LOG | fields actor_process_image_name, event_id )
as edr edr.event_id = edr.event_id
```

Return a maximum of 100 `xdr_data` records with the events of the `agent_id`, `event_id`, and `_product` fields, where the `_product` field is displayed as `product`. The `agent_id`, `event_id`, and `_product` fields are returned from all `xdr_data` records and are then compared to the fields inside the `panw_ngfw_filedata_raw` dataset, where `_time = panw.time`, and the results are added to the new `panw` column. When there is a conflict in the column names between the 2 result sets both columns are used.

```
dataset = xdr_data
| fields agent_id, event_id, _product as product
| join conflict_strategy = both (dataset = panw_ngfw_filedata_raw | fields _product as product)
as panw _time = panw._time
| limit 100
```



### 6.3.13 | limit

#### Abstract

Learn more about the Cortex Query Language **limit** stage that sets the maximum number of records that can be returned in the result set.

#### Syntax

```
limit <number>
```

#### Description

The **limit** stage sets the maximum number of records that can be returned in the result set. To help reduce the Cortex Query Language (XQL) response time, the default results for a Cortex Data Model (XDM) query or an XQL basic query is limited to 1000, when no limit is explicitly stated in the query. This applies to basic queries with no stages except the **fields** stage. This default limit does not apply to widgets, Correlation Rules, public APIs, saved queries, or scheduled queries, where the limit is a maximum of 1,000,000 results.

Using a small limit can greatly increase the performance of your query by reducing the number of records that Cortex AgentiX can return in the result set.

#### Examples

Set the maximum number of records returned by the query to 10.

```
dataset = xdr_data | limit 10
```

### 6.3.14 | replacenull

#### Abstract

Learn more about the Cortex Query Language **replacenull** stage that replaces null field values with a text string.

#### Syntax

```
replacenull <field> = <text string>
```

#### Description

The **replacenull** stage replaces null field values with the specified text string. This guarantees that every field in your result set will contain a value.

If you use the **replacenull** stage, then all subsequent stages that refer to the field's null value must use the replacement text string.

#### Examples

Return the **action\_country** field from every **xdr\_data** records where the **action\_country** field is null, using the text string **N/A** in the place of an empty field value.

```
dataset = xdr_data  
| fields action_country as ac  
| replacenull ac = "N/A"  
| filter ac = "N/A"
```

### 6.3.15 | search

#### Abstract

Learn more about the Cortex Query Language **search** stage that searches for free-text strings.

#### Syntax

```
search "<free_text1>[,<free_text2>, ...]
```

#### Description

The **search** stage searches for free text strings across single or multiple datasets, including all the dataset fields (columns), that are stored in your Cortex AgentiX tenant. This search is a manual process that isn't meant to be included in any features where you can include Cortex Query Language (XQL) queries, such as rules or widgets. Since this search runs on all your data, it can take time for the query to complete.

Search results are presented differently depending on the number of datasets included in the search query:

- Single dataset: All dataset field columns are included in the resulting table.
- Multiple datasets: The resulting table includes a limited number of field columns, specifically the **\_time**, **\_vendor**, **\_product**, **\_dataset**, and **raw\_data** field columns. The **raw\_data** field column includes the JSON with the relevant raw information from the datasets.



- **search** should be the first stage in the query. Only the **config** stage can precede **search**.

- You can refine the search to specify datasets.

Only datasets are supported. You can't refine by preset or search the Cortex Data Model (XDM) schema.

#### **NOTE:**

- If you *do not* specify a dataset in the query, Cortex AgentiX searches all of the existing datasets on your tenant.
- Free text search searches the relevant columns in each dataset. Relevant columns are subject to a change and can vary between datasets.

- When more than one dataset is included in the search, a new column called **raw\_data** is displayed in the Query Results table. This column lists all the fields from the original datasets schema, which you can use to drilldown to specific data in your queries.
- Queries containing **search** do not support the **bin**, **comp**, **top**, or **dedup** stages.
- Queries using the **search** stage are limited to the last 90 days of data. Specifying a time frame outside of this limitation will cause the query to fail.
- Some settings for the free text search are dependent on your configuration of this feature. By default, the standard behavior is followed unless you've requested to disable or enable certain configurations for this feature. Here is a list of the default settings that can be configured:
  - All datasets are included in the search unless you enabled the option to ignore certain datasets.
  - Forensic datasets are not included in the search by default unless you enable the option to include forensic datasets in the free text search. When forensic datasets are configured to be included in the free text search, the forensic datasets are only searched if all datasets in the search command are forensic. This means that forensic datasets are ignored in mixed dataset searches.
  - Snapshots are searched by default unless you enabled the option to ignore snapshots. In addition, the search includes all data in the dataset, across all snapshots, unless you enabled the option to limit the free text search to only search for values in the latest snapshot as defined by the Snapshot SQL.
  - All the rows in the table are searched by default unless you set an XQL **text\_search\_force\_limit\_size** that defines a maximum number of rows per dataset table so only those rows are searched.
  - All JSON fields are searched by default unless you configured the system to skip JSON fields.
  - Any hidden fields configured to be excluded from the correlation dataset are by default not included in the search.

#### Examples

Returns instances of "MacOS" in the **endpoints** dataset.

```
search "MacOs" dataset = endpoints
```

Returns instances of "MacOs" or "failed" in the **endpoints** and **agent\_auditing** datasets.

```
search ª MacOs ª , ª failed ª dataset in (endpoints, agent_auditing)
```

### 6.3.16 | **sort**

#### Abstract

Learn more about the Cortex Query Language **sort** stage that identifies the sort order for records returned in the result set.

#### Syntax

```
sort asc|desc <field1>[, asc|desc <field2>...]
```

#### Description

The **sort** stage identifies the sort order for records returned in the result set. Records can be returned in ascending (**asc**) or descending (**desc**) order. If you include more than one field in the **sort** stage, records are sorted in field specification order.

Keep the following points in mind before running a query with the sort stage:

- To achieve the correct sorting results when a query includes strings representing numbers, it's recommended to sort by integer fields and to convert all string fields to integers; for example, by using the **to\_integer** function.
- When sorting by multiple columns, the sort is saved correctly, but the user interface will only display the results according to the first sorted column.

#### Examples

Return the **action\_boot\_time** and **event\_timestamp** fields from all **xdr\_data** records. Sort the result set first by the **action\_boot\_time** field value in descending order, then by **event\_timestamp** field in ascending order.



```
dataset = xdr_data
| fields action_boot_time as abt, event_timestamp as et
| sort desc abt, asc et
| limit 1
```

### 6.3.17 | Tag

#### Abstract

Learn more about the Cortex Query Language **tag** stage that adds a single tag or list of tags to the `_tag` system field.

#### Syntax

- Add a single tag:

```
| tag add <tag name>
```

- Add a list of tags:

```
| tag add "<tag name1>", "<tag name2>", "<tag name3>",.....
```

#### Description

The **tag** stage is used in combination with the **add** operator to append a single tag or list of tags to the `_tag` system field, which you can easily query in the dataset.

#### Examples

In the `xdr_data` dataset, add a single tag called "test" to the `_tag` system field.

```
dataset = xdr_data
| tag add "test"
```

In the `xdr_data` dataset, add a list of tags, "test1", "test2", and "test3", to the `_tag` system field.

```
dataset = xdr_data
| tag add "test1", "test2", "test3"
```

### 6.3.18 | target

#### Abstract

Learn more about the Cortex Query Language **target** stage that saves query results to a dataset or lookup dataset.

#### Syntax

```
target type=dataset|lookup [append=true|false] <dataset name>
```

#### Description

The **target()** stage saves query results to a named dataset or lookup. These are persistent and can be used in subsequent queries. This stage must be the last stage specified in the query.

The **type** argument defines the type of dataset to create, when a new one needs to be created. The following types are supported:

- **dataset**: A regular dataset of type **USER**. Use **dataset** if you are saving the query results for use in future queries.
- **lookup**: A small lookup table with a 50 MB limit. When uploading a lookup dataset from the Dataset Management page, the limit is 30 MB. This lookup table can be used with parsing rules and downloaded as a JSON file. Use **lookup** if you want to export the query results to a disk.

#### NOTE:

Dataset and lookup tables support low frequency changes of up to 1200 modifications per day. Changes are implemented whenever a dataset or lookup dataset are edited.

#### Optional Append

Use **append** to define whether the data from the current query should be appended to the dataset (**true**) or re-created as a new dataset (**false**). If no **append** is included, the default is **false**. This means that after the query runs the data in an existing dataset is replaced with the new data.

#### IMPORTANT:

When you create or add data to a lookup dataset using the **target** stage, the `_time` field won't be included by default unless you explicitly add it with the **fields** stage.



#### Example 1

Save the results of a simple query to a named dataset.

```
dataset = xdr_data
| fields action_boot_time as abt
| filter abt != null
| target type=dataset abt_dataset
```

Subsequently, you can query the new dataset. Notice that the field names used by the new dataset conform to the aliases that you used when you created the dataset:

```
dataset = abt_dataset
| filter abt = 1603986614040
```

#### Example 2

The following example creates a dataset with the number of agents per country.

```
dataset = xdr_data
| fields agent_id, action_country
| comp count_distinct(agent_id) as count by action_country
| target type=dataset append=false agents_per_country
```

This results in the following XQL JSON:

```
{
  "tables": [
    "xdr_data"
  ],
  "original_query": "\ndataset=xdr_data\n| fields agent_id, action_country\n| comp count_distinct(agent_id) as count by action_country\n| target type=dataset append=false agents_per_country\n",
  "stages": [
    {
      "FIELD_SELECT": {
        "fields": [
          {
            "name": "agent_id",
            "as": None
          },
          {
            "name": "action_country",
            "as": None
          }
        ],
        "exclude": []
      }
    },
    {
      "GROUP": {
        "aggregations": [
          {
            "function": "count_distinct",
            "parameters": [
              "$agent_id"
            ],
            "name": "count"
          }
        ],
        "key": [
          "action_country"
        ]
      }
    }
  ],
  "output": [
    {
      "TARGET": {
        "type": "dataset",
        "target": "agents_per_country",
        "append": False
      }
    }
  ]
}
```

#### 6.3.19 | top

Abstract

Learn more about the Cortex Query Language `top` stage that returns the approximate count of top elements for a field and percentage of the count results.

#### NOTE:

This stage is unsupported with Correlation Rules.



## Syntax

```
top <integer> <field> [by <field1> ,<field2>...] [top_count as <column name>, top_percent as <column name>]
```

## Description

The **top** stage returns the approximate count of top elements for a given field and the percentage of the count results relative to the total number of values for the designated field. Use this top stage to produce approximate results, which are more scalable in terms of memory usage and time.

The **<integer>** in the syntax represents the number of top elements to return. If a number is not specified, up to 10 elements are returned by default. The approximate count is listed in the results table in a column called **TOP\_COUNT** and the percentage in a column called **TOP\_PERCENT**. You can update the column names for both tables by defining **top\_count as <column name>**, **top\_percent as <column name>** in the syntax. If you only define one column name to update in the syntax, the results table displays that column without displaying the other column.

## Examples

Returns a table with 3 columns called **EVENT\_ID**, **TOP\_COUNT**, and **TOP\_PERCENT** with up to 10 unique values for **event\_id** with the corresponding counts and percentages.

```
dataset = xdr_data
| top event_id
```

Returns a table with 3 columns called **ACTION\_COUNTRY**, **EVENT\_ID**, and **TOTAL** with a single unique value for the **event\_id** for each **action\_country** with the corresponding count in the **TOTAL** column.

```
dataset = xdr_data
| top 1 event_id by action_country top_count as total
```

## 6.3.20 | transaction

### Abstract

Learn more about the Cortex Query Language transaction stage used to find transactions based on events that meet certain constraints.

#### NOTE:

This stage is unsupported with RT Correlation Rules.

## Syntax

```
transaction <field_1, field_2, ...> [span = <time> [timeshift = <epoch time> [timezone = "<time zone>"]] | startswith = <condition> endswith = <condition>
allowunclosed= true|false] maxevents = <number of events per transaction>
```

## Description

The **transaction** stage is used to find transactions based on events that meet certain constraints. This stage aggregates all fields in a JSON string array by fields defined as transaction fields. For example, using the **transaction** stage to find transactions based on the **user** and **user\_ip** fields will make the aggregation of json strings of all fields by the **user** and **user\_ip** fields. A maximum of 50 fields can be aggregated in a **transaction** stage.

You can also configure whether the transactions falls within a certain time frame, which is optional to define. You can set one of the following:

- **span=<time>**: Use this command to set a time frame per transaction, where **<time>** is a combination of a number and time suffix. Set one time suffix from the list of available options listed in the table below. In addition, you can define a particular start time for grouping the events in your query according to the Unix epoch time by setting **timeshift = <epoch time> timezone = "<time zone>"**, which are both optional. You can configure the **<time zone>** offset using an hours offset, such as **+08:00**, or using a time zone name from the List of Supported Time Zones, such as **"America/Chicago"**. The query still runs without defining the epoch time or time zone. If no **timeshift = <epoch time> timezone = "<time zone>"** is set, the query runs according to last time set in the log.
- **startswith** and **endswith**: Use these commands to set a condition for the beginning or end of the transaction, where the condition can be a logical expression or free text search.

Set the **allowunclosed** flag to **true** to include transactions which don't contain an ending event. The last event will be 12 hours after the starting event. By default, this is set to **true** and transactions without an ending event are included.

Use the **maxevents** command to define the maximum number of events to include per transaction. If this command is not set, the default value is 100.

When using the transaction stage, 5 additional fields are added to the results displayed:



- `_start_time`: Indicates the initial timestamp of the transaction.
- `_end_time`: Indicates the last timestamp for the transaction.
- `_duration`: Displays the difference in seconds between the timestamps for the first and last events in the transaction.
- `_num_of_rows`: Indicates the number of events in the transaction.
- `_transaction_id`: Displays the unique transaction ID.

Time Suffix	Description
MS	milliseconds
S	seconds
M	minutes
H	hours
D	days
W	weeks
MO	months
Y	years

#### Example using Span

Return a maximum of 10 events per transaction from the `xdr_data` records based on the `user` and `agent_id` fields, where the transaction time frame is 1 hour.

```
dataset=xdr_data
|transaction user, agent_id span=1h timeshift = 1615353499
timezone = +08:00 maxevents=10
```

This query results in the following XQL JSON:

```
{"TRANSACTION": {"fields": ["user", "agent_id"], "maxevents": 10, "span": {"amount": 1, "units": "h", "timeshift": None}}}
```

#### Example using Startswith and Endswith

Return a maximum of 99 events per transaction from the `xdr_data` records based on the `f1` and `f2` fields. The starting event of each transaction is an event, where one of the fields contains a string "`str_1`", and the ending event of each transaction is an event, where one of the fields contains a string "`str_2`".

```
dataset=xdr_data
| transaction f1, f2 startswith="str_1" endswith="str2" maxevents=99
```

This query results in the following XQL JSON:

```
{"TRANSACTION": {"fields": ["f1", "f2"], "search": {"startswith": {"filter": {"free_text": "str_1"}}, "endswith": {"filter": {"free_text": "str2"}}, "maxevents": 99}}
```

### 6.3.21 | union

#### Abstract

Learn more about the Cortex Query Language `union` stage that combines two result sets into a single result set.

#### Syntax

```
union <datasetname>
```



```
union (<inner xql query>)
```

#### Description

The `union()` stage combines two result sets into one result. It can be used in two different ways.

If a dataset name is provided with no other arguments, the two datasets are combined for the duration of the query, and the fields in both datasets are available to subsequent stages.

If a Cortex Query Language (XQL) query is provided to this stage, the result set from that XQL union query is combined with the result set from the rest of the query. This is effectively an inner join statement.

#### Examples

First, create a dataset using the target stage. This results in a persistent stage that we can use later with a `union` stage.

```
dataset = xdr_data
| filter event_type = FILE and event_sub_type = FILE_WRITE
| fields agent_id, action_file_sha256 as file_hash, agent_hostname
| target type=dataset file_event
```

Then run a second query, using `union` so that the query can access the contents of the `file_event` dataset. Notice that this second query uses the `file_hash` alias that was defined for the `file_event` dataset.

```
dataset = xdr_data
| filter event_type = PROCESS and event_sub_type = PROCESS_START
| union file_event
| fields agent_id, agent_hostname, file_hash,
  actor_process_image_path as executed_by,
  actor_process_signature_vendor as executor_signer
| filter file_hash != null and executed_by != null
```

### 6.3.22 | view

#### Abstract

Learn more about the Cortex Query Language `view` stage that configures the display of the result set.

#### Syntax

```
view highlight fields = <field1>[,<field2>,...] values = <value1>[,<value2>,...]
view graph type = column|line|pie xaxis = <field1>
  yaxis = <field2> [<optional parameters>]
  [series = <field3> [<optional parameters>] ]
view column order = default|populated
```

#### Description

The `view()` stage configures the display of the result set in the following ways:



- **highlight**: Highlights specified strings that Cortex AgentIX finds on specified fields. The highlight values that you provide are performed as a substring search, so only partial value can be highlighted in the final results table.
- **graph type**: Creates an **area**, **bubble**, **column**, **funnel**, **gauge**, **line**, **map**, **pie**, **scatter**, **single**, or **wordcloud** chart based on the values found for the fields specified in the **xaxis** and **yaxis** parameters. In this mode, **view** also offers a large number of parameters that allow you to control colors, decorations, and other behavior used for the final chart, where the options can differ depending on the type of graph selected. You can also define a graph **subtype**, when setting the **graph type** to either **column** or **pie**.
  - (Optional) **series**: When creating an **area**, **bubble**, **column**, **line**, **map**, or **scatter** chart, you can define a **series** parameter by specifying a field (column) to group chart results based on y-axis values. The series parameter is only supported when defining a single y-axis value.

You can also generate graphs and outputs of your query data directly in the Query Builder after running a Cortex Query Language (XQL) query in the Query Results tab without having to add the syntax in the query. For more information, see Graph query results.

#### **NOTE:**

If you use **graph type**, the fields specified for **xaxis** and **yaxis** must be collatable or the query will fail.

- **column order**: Enables you to list the query results by popularity, where the most non-null returned fields are displayed first using the syntax **view column order = populated**. By default, if **column order** is not defined (or **view column order=default**), the original column order is used.

#### **NOTE:**

This option does not apply to Cortex Query Language (XQL) queries in widgets, Correlation Rules, public APIs, reports, and dashboards. If you include the **view column order** syntax in these types of queries, Cortex AgentIX disregards the stage from the query and completes the rest of the query.

#### Examples

Use the dedup stage collect unique combinations of **event\_type** and **event\_sub\_type** values. Highlight the word "STREAM" when it appears in the result set.

```
dataset = xdr_data
| fields event_type, event_sub_type
| dedup event_type, event_sub_type by asc _time
| view highlight fields = event_sub_type values = "STREAM"
```

Count the number of unique files accessed by each user, and show a column graph of the results, where the number of unique files are grouped by username. This query uses comp count\_distinct to calculate the number of unique files per username.

```
dataset = xdr_data
| fields actor_effective_username as username, action_file_path as file_path
| filter file_path != null and username != null
| comp count_distinct(file_path) as file_count by username
| view graph type = column xaxis = username yaxis = file_count series = username
```

Count the number of unique files accessed by each user, and display the results by popularity according to the most non-null values returned fields. This query uses comp count\_distinct to calculate the number of unique files per username.

```
dataset = xdr_data
| fields actor_effective_username as username, action_file_path as file_path
| filter file_path != null and username != null
| comp count_distinct(file_path) as file_count by username
| view column order = populated
```

### 6.3.23 | **windowcomp**

#### Abstract

Learn more about the Cortex Query Language **windowcomp** stage that precedes functions calculating statistics.

#### Syntax

```
windowcomp <analytic function> (<field>){by <fieldA> [,<fieldB>,...]} [sort [asc|desc] <field1> [, [asc|desc] <field2>,...]] [between 0|null|<number>|-<number>]
[and 0|null|<number>|-<number>] [frame_type=range]] [as <alias>]
```

#### **NOTE:**

Defining a field with an analytic function is optional when using a count function. For rank and row\_number functions, it's not allowed.

#### Description

The **windowcomp** stage precedes functions calculating statistics. The results compute values over a group of rows and return a single result for each row, for all records that contain matching values for the fields identified using a combination of the by clause, sort, and range. Only one function can be defined per field, while the other parameters are optional. Yet, it's possible to define multiple fields.

#### Example 113.

```
| windowcomp sum(field_1) by field_2 sort field_3 as field_4, min(field_5) by field_6 sort field_7 as field_8
```



## Supported functions

This stage includes the following functions:

Function Type	Function
Numbering functions	<ul style="list-style-type: none"> <li>rank</li> <li>row_number</li> </ul>
Navigation functions	<ul style="list-style-type: none"> <li>first_value</li> <li>lag</li> <li>last_value</li> </ul>
Statistical aggregate functions	<ul style="list-style-type: none"> <li>stddev_sample</li> <li>stddev_population</li> </ul>
Aggregate functions	<ul style="list-style-type: none"> <li>avg</li> <li>count</li> <li>max</li> <li>median</li> <li>min</li> <li>sum</li> </ul>

## Optional parameters

The optional parameters available to define in the `windowcomp` function are explained in the following table:

Optional Parameters	Syntax	Description
By clause	<code>[by &lt;fieldA&gt; [, &lt;fieldB&gt;, ...]]</code>	<p>The <code>by</code> clause is used to break up the input field rows into separate partitions, over which the <code>windowcomp</code> function is independently evaluated.</p> <ul style="list-style-type: none"> <li>Multiple partition fields are allowed when using a partition <code>by</code> clause.</li> <li>When this optional clause is omitted, all rows in the input table comprise a single partition.</li> </ul>
Sort	<code>[sort [asc desc] &lt;field1&gt; [, [asc desc] &lt;field2&gt;, ...]]</code>	Defines how field rows are ordered within a partition as either ascending ( <code>asc</code> ) or descending ( <code>desc</code> ). This clause is optional in most situations, but is required in some cases for navigation functions and <code>rank</code> function.



Optional Parameters	Syntax	Description
Between window frame clause	<pre>[between 0 null  &lt;number&gt; -&lt;number&gt; [and 0 null &lt;number&gt; -&lt;number&gt;]</pre>	<p>Sets the window frame around the current row within a partition, over which the window function is evaluated. Numbering functions and the <code>lag</code> function can't be used in the window frame clause. Creates a window frame with a lower and upper boundary. The first boundary represents the lower boundary. The second boundary represents the upper boundary. Every boundary can include the following options:</p> <ul style="list-style-type: none"> <li>• <code>null</code>: Starts at the beginning or at the end of the partition, depending on the placement of the <code>null</code>.</li> <li>• <code>0</code>: Is set to the current row, where the window frame starts or ends at the current row.</li> <li>• positive/negative <code>&lt;number&gt;</code>: The end of the window frame or the start of the window frame relative to the current row. <ul style="list-style-type: none"> <li>◦ If only a start <code>&lt;number&gt;</code> is defined, only a negative number is allowed: <code>-&lt;number&gt;</code>.</li> <li>◦ If a start <code>&lt;number&gt;</code> and end <code>&lt;number&gt;</code> are defined, the end <code>&lt;number&gt;</code> must be greater than the start <code>&lt;number&gt;</code>.</li> </ul> </li> </ul> <p>If the <code>sort</code> is included, but the window frame clause isn't, the following window frame clause is used by default:</p> <pre>between null and 0</pre>
frame_type	<pre>[frame_type=rows  range]</pre>	<p>Defines the option of the frame as either:</p> <ul style="list-style-type: none"> <li>• <code>rows</code> (default): Computes the window frame based on physical offsets from the current row. For example, you could include two rows before and after the current row. To apply the default <code>frame_type=rows</code>, nothing needs to be added to the <code>windowcomp</code> stage syntax as it's automatically built into the query.</li> <li>• <code>range</code>: Computes the window frame based on a logical range of rows around the current row, based on the current row's sort key value. The provided range value is added or subtracted to the current row's key value to define a starting or ending range boundary for the window frame. Setting the <code>range</code> with start or end numeric, nonzero boundaries requires using exactly one numeric type of sort field.</li> </ul> <p>When setting <code>frame_type=range</code>, the <code>sort</code> must be included in the <code>windowcomp</code> stage syntax; otherwise, only <code>between null and null</code> is supported.</p> <p>Example 114.</p> <p>This is unsupported:</p> <pre>  windowcomp sum(field_a) between -2 and 0 frame_type = range</pre> <p>Yet, the following is supported:</p> <pre>  windowcomp sum(field_a) sort desc field_b between -1 and 1 frame_type = range</pre> <p>Or</p> <pre>  windowcomp sum(field_a) between null and null frame_type = range</pre>
Alias clause	<pre>[as &lt;alias&gt;]</pre>	<p>Use the <code>alias</code> clause to provide a column label (field name) for the <code>windowcomp</code> results. When the new field name already exists in the schema, it's replaced with the new name.</p> <p>Example 115.</p> <p>If the <code>xdr_data</code> dataset already has a field in the schema called <code>existing_field</code>, the new <code>existing_field</code> replaces the old one.</p> <pre>dataset = xdr_data   windowcomp sum(field_a) as existing_field</pre>



## Examples

Data table for ips dataset

The examples provided are based on the following data table for a dataset called `ips`:

Ip	Category	Logins
192.168.10.1	pc	23
192.168.10.2	server	2
192.168.20.1	pc	9
192.168.20.4	server	8
192.168.20.5	pc	2
192.168.30.1	pc	10

Query 1: Compute the total logins for all IPs

```
dataset = ips
| windowcomp sum(logins) as total_logins
```

Output results table

Ip	Logins	Category	Total_logins
192.168.10.2	2	server	54
192.168.20.5	2	pc	54
192.168.20.4	8	server	54
192.168.20.1	9	pc	54
192.168.30.1	10	pc	54
192.168.10.1	23	pc	54

Query 2: Compute a subtotal for each category

```
dataset = ips
| windowcomp sum(logins) by category sort asc logins between null and null as total_logins
```

Output results table

Ip	Logins	Category	Total_logins
192.168.10.2	2	server	10



Ip	Logins	Category	Total_logins
192.168.20.4	8	server	10
192.168.20.5	2	pc	44
192.168.20.1	9	pc	44
192.168.30.1	10	pc	44
192.168.10.1	23	pc	44

Query 3: Compute a cumulative sum for each category

The sum is computed with respect to the order defined using the `sort` clause. These two queries produce the same results:

```
dataset = ips
| windowcomp sum(logins) by category sort asc logins between null and 0 as total_logins
```

OR

```
dataset = ips
| windowcomp sum(logins) by category sort asc logins between null as total_logins
```

Output results table

Ip	Logins	Category	Total_logins
192.168.10.2	2	server	2
192.168.20.4	8	server	10
192.168.20.5	2	pc	2
192.168.20.1	9	pc	11
192.168.30.1	10	pc	21
192.168.10.1	23	pc	44

Query 4: Compute a cumulative sum, where only preceding rows are analyzed.

The analysis starts two rows before the current row in the partition.

```
dataset = ips
| windowcomp sum(logins) sort asc logins between null and -2 as total_logins
```

Output results table

Ip	Logins	Category	Total_logins
192.168.10.2	2	server	NULL
192.168.20.5	2	pc	NULL



Ip	Logins	Category	Total_logins
192.168.20.4	8	server	2
192.168.20.1	9	pc	4
192.168.30.1	10	pc	12
192.168.10.1	23	pc	21

Query 5: Compute a changing average

The lower boundary is 1 row before the current row. The upper boundary is 1 row after the current row.

```
dataset = ips
| windowcomp avg(logins) sort asc logins between -1 and 1 as avg_logins
```

Output results table

Ip	Logins	Category	Avg_logins
192.168.10.2	2	server	2
192.168.20.5	2	pc	4
192.168.20.4	8	server	6.33333
192.168.20.1	9	pc	9
192.168.30.1	10	pc	14
192.168.10.1	23	pc	16.5

Query 6: Retrieve the most popular IP in each category

Defines how rows in a window are partitioned and ordered in each partition.

```
dataset = ips
| windowcomp last_value(ip) by category sort asc logins between null and null as most_popular
```

Output results table

Ip	Logins	Category	Most_popular
192.168.10.2	2	server	192.168.20.4
192.168.20.4	8	server	192.168.20.4
192.168.20.5	2	pc	192.168.10.1
192.168.20.1	9	pc	192.168.10.1



<b>Ip</b>	<b>Logins</b>	<b>Category</b>	<b>Most_popular</b>
192.168.30.1	10	pc	192.168.10.1
192.168.10.1	23	pc	192.168.10.1

Query 7: Calculate the rank of each IP within the category based on the login

```
dataset = ips
| windowcomp rank() by category sort asc logins as rank
```

Output results table

<b>Ip</b>	<b>Logins</b>	<b>Category</b>	<b>Rank</b>
192.168.10.2	2	server	1
192.168.20.4	8	server	2
192.168.20.5	2	pc	1
192.168.20.1	9	pc	2
192.168.30.1	10	pc	3
192.168.10.1	23	pc	4

Query 8: Retrieve the most popular IP in a specific window frame by range and not category

The window frame analyzes up to three rows at a time.

```
dataset = ips
| windowcomp last_value(ip) by category sort asc logins between -1 and 1 as most_popular
```

Output results table

<b>Ip</b>	<b>Logins</b>	<b>Category</b>	<b>Most_popular</b>
192.168.10.2	2	server	192.168.20.4
192.168.20.4	8	server	192.168.20.4
192.168.20.5	2	pc	192.168.20.1
192.168.20.1	9	pc	192.168.30.1
192.168.30.1	10	pc	192.168.10.1
192.168.10.1	23	pc	192.168.10.1

Query 9: Retrieve the number of IPs that have similar logins



Count in range of -1 and 1 from their login value.

```
dataset = ips | fields ip, category , logins  
| windowcomp count() sort asc logins between -1 and 1 frame_type = range as similar_logins
```

Output results table

Ip	Logins	Category	Similar_logins
192.168.10.5	2	pc	2
192.168.10.2	2	server	2
192.168.20.4	8	server	2
192.168.20.1	9	pc	3
192.168.30.1	10	pc	2
192.168.10.1	23	pc	1

## 6.4 | Functions

Abstract

Learn more the functions that can be used with Cortex Query Language (XQL) stages in Cortex AgentIX.

Some Cortex Query Language (XQL) stages can call XQL functions to convert the data to a desired format. For example, the `current_time()` function returns the current timestamp, while the `extract_time()` function can obtain the hour information in the timestamp. Functions may or may not need input parameters. The `filter` and `alter` stages are the two stages that can use functions for data transformations.

### 6.4.1 | add

Abstract

Learn more about the Cortex Query Language `add()` function that adds two integers.

Syntax

```
add (<string> | <integer>, <string> | <integer>)
```

Description

The `add()` function adds two positive integers. Parameters can be either integer literals, or integers as a string type, such as might be contained in a data field.

Example

```
dataset = xdr_data  
| alter mynum = add(action_file_size, 3)  
| fields action_file_size, mynum  
| filter action_file_size > 0  
| limit 1
```

### 6.4.2 | approx\_count

Abstract

Learn more about the Cortex Query Language `approx_count` approximate aggregate comp function.

Syntax

```
comp approx_count(<field>) [as <alias>] [by <field1>[,<field2>...]] [addrwdata = true|false [as <target field>]]
```



## Description

The **approx\_count** approximate aggregate is a comp function that counts the number of distinct values in the given field over a group of rows. For the group of rows, the function returns an approximate result as a single integer value, for all records that contain matching values for the fields identified in the **by** clause. Use this approximate aggregate function to produce approximate results, instead of exact results used with regular aggregate functions, which are more scalable in terms of memory usage and time. This approximate aggregate function is used in combination with a **comp** stage.

In addition, you can configure whether the raw data events are displayed by setting **addradwdata** to either **true** or **false** (default), which are used to configure the final **comp** results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

## Example

Returns a single integer value after approximately counting the number of distinct values in the **event\_id** field over a group of rows.

```
dataset = xdr_data
| fields event_id
| comp approx_count(event_id)
```

### 6.4.3 | approx\_quantiles

#### Abstract

Learn more about the Cortex Query Language **approx\_quantiles** approximate aggregate comp function.

#### Syntax

```
comp approx_quantiles(<field>, <number>, <true/false>) [as <alias>] [by <field1>[,<field2>...]][addradwdata = true|false [as <target field>]]
```

#### Description

The **approx\_quantiles** approximate aggregate is a comp function returns the approximate boundaries as a single value for a group of distinct or non-distinct values (default **false**) for the specified field over a group of rows, for all records that contain matching values for the fields identified in the **by** clause. This function returns an array of **<number> + 1** elements, where the first element is the approximate minimum and the last element is the approximate maximum. Use this approximate aggregate function to produce approximate results, instead of exact results used with regular aggregate functions, which are more scalable in terms of memory usage and time. This approximate aggregate function is used in combination with a **comp** stage.

In addition, you can configure whether the raw data events are displayed by setting **addradwdata** to either **true** or **false** (default), which are used to configure the final **comp** results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

#### Examples

##### Distinct Values Example

Returns the approximate boundaries for a group of distinct values in the **event\_id** field.

```
dataset = xdr_data
| fields event_id
| comp approx_quantiles(event_id, 100, true)
```

##### Non-Distinct Values Example

Returns the approximate boundaries for a group of non-distinct values in the **event\_id** field.

```
dataset = xdr_data
| fields event_id
| comp approx_quantiles(event_id, 100)
```

### 6.4.4 | approx\_top

#### Abstract

Learn more about the Cortex Query Language **approx\_top** approximate aggregate comp function.

#### Syntax

```
comp approx_top as count

comp approx_top(<string field>, <number>) [as <alias>] [by <field1>[,<field2>...]][addradwdata = true|false [as <target field>]]

comp approx_top as sum

comp approx_top(<string field>, <number>, <weight string field>) [as <alias>] [by <field1>[,<field2>...]][addradwdata = true|false [as <target field>]]
```



## Description

The **approx\_top** approximate aggregate is a comp function that, depending on the number of parameters, returns either an approximate count or sum of top elements. This approximate aggregate function returns a single value for the given field over a group of rows, for all records that contain matching values for the fields identified in the **by** clause. This function is used in combination with a **comp** stage. When a third parameter is specified, it references a field that contains a numeric value (weight) that is used to calculate a sum. The return value is an array with up to <number> of JSON strings. Each string represents an object (struct) containing 2 keys and corresponding values. The keys depend on whether a third parameter has been supplied or not.

When defining **approx\_top** to count and the third parameter is omitted, each struct will have these keys: "value" and "count", where the "value" specifies a unique field value and "count" specifies the number of occurrences. When the third parameter is specified in **approx\_top**, it has to be a name of a field that contains a numeric value that is used to calculate the final sum for each unique value in the first specified field. Each struct in this case will have these keys: "value" and "sum".

In addition, you can configure whether the raw data events are displayed by setting **addrawdata** to either **true** or **false** (default), which are used to configure the final **comp** results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

Use this approximate aggregate function to produce approximate results, instead of exact results used with regular aggregate functions, which are more scalable in terms of memory usage and time.

## Examples

```
comp approx_top as count
```

Returns an approximate count of the top 10 agent IDs in the **agent\_id** field that appear the most frequently. The return value is an array containing 10 JSON strings with a "value" and "count".

```
dataset = xdr_data
| fields agent_id
| comp approx_top(agent_id, 10)
```

```
comp approx_top as sum
```

Returns an approximate sum of the top 10 agent IDs in the **agent\_id** field by their **action\_session\_duration**. The return value is an array containing 10 JSON strings with a "value" and "sum" for each **agent\_id**.

```
dataset = xdr_data
| fields agent_id, action_session_duration
| comp approx_top(agent_id, 10, action_session_duration)
```

## 6.4.5 | array\_all

### Abstract

Learn more about the Cortex Query Language **array\_all()** function.

### Syntax

```
array_all(<array>, "@element"<operator>"<array element>")
```

#### NOTE:

The <operator> can be any of the ones supported, such as = and !=.

## Description

The **array\_all()** function returns **true** when all the elements in a particular array match the condition in the specified array element. Otherwise, the function returns **false**.

### Example

When the **dfe\_labels** array is not empty, use the alter stage to create a new column called **x** that returns true when all the elements in the **dfe\_labels** array is equal to **network**; otherwise, the function returns **false**.

```
dataset = xdr_data
| filter dfe_labels != null
| alter x = array_all(dfe_labels , "@element" = "network")
| fields x, dfe_labels
| limit 100
```

## 6.4.6 | array\_any

### Abstract

Learn more about the Cortex Query Language **array\_any()** function.



## Syntax

```
array_any(<array>, "@element"<operator>"<array element>")
```

### NOTE:

The <operator> can be any of the ones supported, such as = and !=.

## Description

The **array\_any()** function returns **true** when at least 1 element in a particular array matches the condition in the specified array element. Otherwise, the function returns **false**.

## Example

When the **dfe\_labels** array is not empty, use the alter stage to create a new column called **x** that returns true when at least 1 element in the **dfe\_labels** array is equal to **network**; otherwise, the function returns **false**.

```
dataset = xdr_data
| filter dfe_labels != null
| alter x = array_any(dfe_labels , "@element" = "network")
| fields x, dfe_labels
| limit 100
```

## 6.4.7 | arrayconcat

### Abstract

Learn more about the Cortex Query Language **arrayconcat()** function that returns an array containing unique values found in the original array.

## Syntax

```
arrayconcat (<array1>,<array2>[,<array3>...])
```

## Description

The **arrayconcat()** function accepts two or more arrays, and it joins them into a single array.

## Example

Given three arrays:

```
first_array : [1,2,3]
second_array : [44,55]
third_array : [4,5,6]
```

Using this query:

```
alter all_arrays = arrayconcat(first_array, second_array, third_array)
```

Results in an **all\_arrays** field containing:

```
[1,2,3,44,55,4,5,6]
```

## 6.4.8 | arraycreate

### Abstract

Learn more about the Cortex Query Language **arraycreate()** function that returns an array based on the given parameters defined for the array elements.

## Syntax

```
arraycreate ("<array element1>", "<array element2>","",...)
```

## Description

The **arraycreate()** function returns an array based on the given parameters defined for the array elements.

## Example

Returns a final array to a field called **x** that is comprised of the elements [1,2].

```
dataset = xdr_data
| alter x = arraycreate("1", "2")
| fields x
```



## 6.4.9 | arraydistinct

### Abstract

Learn more about the Cortex Query Language `arraydistinct()` function that returns an array containing unique values found in the original array.

### Syntax

```
arraydistinct (<array>)
```

### Description

The `arraydistinct()` function accepts an array, and it returns a new array containing only unique elements found in the original array. That is, given the array:

```
[0,1,1,1,4,5,5]
```

This function returns:

```
[0,1,4,5]
```

## 6.4.10 | arrayfilter

### Abstract

Learn more about the Cortex Query Language `arrayfilter()` function.

### Syntax

```
arrayfilter(<array>, <condition>)
arrayfilter(<array>, "@element"<operator>"<array element>")
```

#### NOTE:

The `<operator>` can be any of the ones supported, such as `=` and `!=`.

### Description

The `arrayfilter()` function returns a new array with the elements which meet the given condition. The function does this by filtering the results of an array in one of the following ways:

- Returns the results when a certain condition is applied to the array.
- Returns the results when a particular array is set to a specified array element.

Though it's possible to define the `arrayfilter()` function with any condition, the examples below focus on conditions using the `@element` that are based on the current element being tested.

### Basic Example

When the `dfe_labels` array is not empty, use the alter stage to assign a value to a field called `x` that returns the value of the `arrayfilter` function. The `arrayfilter` function filters the `dfe_labels` array for the array element set to `network`.

```
dataset = xdr_data
| filter dfe_labels != null
| alter x = arrayfilter(dfe_labels , "@element" = "network")
| fields x, dfe_labels
| limit 100
```

### Advanced Example

This queries below illustrate how to check whether any IPs are included or not included in the blocked list called CIDRS. The Query Results tables are also included to help explain what happens as the `arrayfilter()` function is slightly modified.

#### Return Non-Matching CIDRS

This query returns results for each IP that don't match anything in the CIDRS array blocked list:

```
dataset = xdr_data
| limit 1
| alter cidrs = arraycreate("10.0.0.0/8", "172.16.0.0/16"), ip = arraycreate("192.168.1.1", "172.16.20.18")
| fields cidrs, ip
| arrayexpand ip
| alter non_matching_cidrs = arrayfilter(cidrs, ip not incidr "@element")
```

### Results:

The following table details for each IP the logic that is first performed before the final results for the query are displayed:



IP	Statement	TRUE/FALSE
192.168.1.1	not in 10.0.0.0/8	TRUE
192.168.1.1	not in 172.16.0.0/16	TRUE
172.16.20.18	not in 10.0.0.0/8	TRUE
172.16.20.18	not in 172.16.0.0/16	FALSE

For each IP, an array of CIDRS is returned in the NON\_MATCHING\_CIDRS column, which doesn't match the CIDRS array. In addition, from the above table, `arrayfilter()` only returns anything that resolves as TRUE. This explains the query results displayed in the following table:

IP	CIDRS	NON_MATCHING_CIDRS
192.168.1.1	10.0.0.0/8,172.16.0.0/16	10.0.0.0/8,172.16.0.0/16
172.16.20.18	10.0.0.0/8,172.16.0.0/16	10.0.0.0/8

Return Matching CIDRS

Now, let's update the query to return results for each IP that match anything in the CIDRS array:

```
dataset = xdr_data
| limit 1
| alter cidrs = arraycreate("10.0.0.0/8","172.16.0.0/16"), ip = arraycreate("192.168.1.1", "172.16.20.18")
| fields cidrs, ip
| arrayexpand ip
| alter matching_cidrs = arrayfilter(cidrs, ip incidr "@element")
```

Results:

The following table details for each IP the logic that is first performed before the final results for the query are displayed:

IP	Statement	TRUE/FALSE
192.168.1.1	in 10.0.0.0/8	FALSE
192.168.1.1	in 172.16.0.0/16	FALSE
172.16.20.18	in 10.0.0.0/8	FALSE
172.16.20.18	in 172.16.0.0/16	TRUE

For each IP, an array of CIDRS is returned in the MATCHING\_CIDRS column, which matches the CIDRS array. In addition, from the above table, `arrayfilter()` only returns anything that resolves as TRUE. This explains the query results displayed in the following table:

IP	CIDRS	MATCHING_CIDRS
192.168.1.1	10.0.0.0/8,172.16.0.0/16	empty array



IP	CIDRS	MATCHING_CIDRS
172.16.20.18	10.0.0.0/8,172.16.0.0/16	172.16.0.0/16

#### 6.4.11 | arrayindex

##### Abstract

Learn more about the Cortex Query Language `arrayindex()` function that returns the array element contained at the specified index.

##### Syntax

```
arrayindex(<array>, <index>)
```

##### Description

The `arrayindex()` function returns the value contained in the specified array position. Arrays are 0-based, and negative indexing is supported.

##### Examples

Use the split function to split IP addresses into an array of octets. Return the 3rd octet contained in the IP address.

```
dataset = xdr_data
| fields action_local_ip as alii
| alter ip_third_octet = arrayindex(split(alii, ".") , 2)
| filter alii != null and alii != "0.0.0.0"
| limit 10
```

#### 6.4.12 | arrayindexof

##### Abstract

Learn more about the Cortex Query Language `arrayindexof()` function that returns the index value of an array.

##### Syntax

```
arrayindexof(<array>, <condition>)
arrayindexof(<array>, "@element"<operator>"<array_element>")
```

##### NOTE:

The `<operator>` can be any of the ones supported, such as `=` and `!=`.

##### Description

The `arrayindexof()` function enables you to return a value related to an array in one of the following ways.

- Returns 0 if a particular array is not empty and the specified condition is true. If the condition is not met, a NULL value is returned.
- Returns the 0-based index of a particular array element if a particular array is not empty and the specified condition using an `@element` is true. If the condition is not met, a NULL value is returned.

##### Examples

##### Condition

Use the alter stage to assign a value returned by the `arrayindexof` function to a field called `x`. The `arrayindexof` function reviews the `dfe_labels` array and returns 0 if the array is not empty and the `backtrace_identities` array contains more than 1 element. Otherwise, a NULL value is assigned to the `x` field.

```
dataset in (xdr_data)
| alter x = arrayindexof(dfe_labels , array_length(backtrace_identities) > 1)
| fields x, dfe_labels
| limit 100
```

##### @Element

When the `dfe_labels` array is not empty, use the alter stage to assign the 0-based index value returned by the `arrayindexof` function to a field called `x`. The `arrayindexof` function reviews the `dfe_labels` array and looks for the array element set to `network`. Otherwise, a NULL value is assigned to the `x` field.



```
dataset = xdr_data
| filter dfe_labels != null
| alter x = arrayindexof(dfe_labels , "@element" = "network")
| fields x, dfe_labels
| limit 100
```

#### 6.4.13 | `array_length`

##### Abstract

Learn more about the Cortex Query Language `array_length()` function that returns the length of an array.

##### Syntax

```
array_length (<array>)
```

##### Description

The `array_length()` function returns the number of elements in an array. When `array_length` receives an empty field the result returned is NULL.

##### Example

```
dataset = xdr_data
| fields action_local_ip as alii
| alter ip_len = array_length(split(alii, "."))
| filter alii != null and alii != "0.0.0.0"
| limit 1
```

#### 6.4.14 | `arraymap`

##### Abstract

Learn more about the Cortex Query Language `arraymap()` function that applies a callable function to every element of an array.

##### Syntax

```
arraymap (<array>, <function()>)
```

##### Description

The `arraymap()` function applies a specified function to every element of an array. For functions that require afieldname, use "`@element`".

##### Examples

Extract the MAC address from the `agent_interface_map` field. This example uses the `json_extract_scalar`, `to_json_string`, `json_extract_array`, and `arraystring` functions to extract the desired information.

```
dataset = xdr_data
| alter mac =
  arraystring (
    arraymap (
      json_extract_array (to_json_string(agent_interface_map), "$."),
      json_extract_scalar ("@element", "$.mac")
    ), ",")
```

#### 6.4.15 | `arraymerge`

##### Abstract

Learn more about the Cortex Query Language `arraymerge()` function that returns an array created from a merge of the inner json-string arrays.

##### Syntax

```
arraymerge(<field>)
```

##### Description

The `arraymerge()` function returns an array, which is created from a merge of the inner json-string arrays, including merging a number of `arraymap()` function arrays. This function accepts a single array of json-strings, which is the `<field>` in the syntax.

##### Example 1

Returns a final array called `result` that is created from a merge of the inner json-string arrays from array `x` and array `y` with the values `["a", "b", "c", "d"]`.



```
dataset = xdr_data
| alter x= to_json_string(arraycreate("a","b")), y = to_json_string(arraycreate("c","d"))
| alter xy = arraycreate(x,y)
| alter xy=arraymerge(xy)
```

#### Example 2

Returns a final array that is created from a merge of the arraymap by extracting the IP address from the agent\_interface\_map field and the first IPV4 address found in the first element of the `agent_interface_map` array. This example uses the `to_json_string` and `json_extract_array` functions to extract the desired information.

```
dataset = xdr_data
| alter a =
arraymerge (arraymap (agent_interface_map, to_json_string (json_extract_array (to_json_string("@element"), "$.ipv4") ) ) )
```

### 6.4.16 | arrayrange

#### Abstract

Learn more about the Cortex Query Language `arrayrange()` function that returns a portion of an array based on specified array indices.

#### Syntax

```
arrayrange (<array>, <start>, <end>)
```

#### Description

The `arrayrange()` function returns a portion, or a slice, of an array given a start and end range. Indices are 0-based, and the start range is inclusive, but the end range is exclusive.

#### Example

So if you have an array:

```
[0,1,2,3,4,5,6]
```

and you specify:

```
arrayrange(<array>, 2, 4)
```

the function will return:

```
[2,3]
```

If you specify an end index that is higher than the last element in the array, the resulting array contains the starting element to the end of the array.

```
arrayrange(<array>, 2, 8)
```

The function will return:

```
[2,3,4,5,6]
```

### 6.4.17 | arraystring

#### Abstract

Learn more about the Cortex Query Language `arraystring()` function that returns a string from an array, where each array element is joined by a defined delimiter.

#### Syntax

```
arraystring (<string>, <delimiter>)
```

#### Description

The `arraystring()` function returns a string from an array, where each array element is joined by a defined delimiter.

#### Examples

Retrieve all `action_app_id_transitions` that are not null, combine each array into a string where array elements are delimited by ":", and then use dedup the resulting string.

```
dataset = xdr_data
| fields action_app_id_transitions as aait
| alter transitions_string = arraystring(aait, " : ")
| dedup transitions_string by asc _time
| filter aait != null
```



## 6.4.18 | avg

### Abstract

Learn more about the Cortex Query Language **avg** used with both **comp** and **windowcomp** stages.

### Syntax

#### comp stage

```
comp avg(<field>) [as <alias>] by <field_1>,<field_2> [addraddata = true|false [as <target field>]]
```

#### windowcomp stage

```
windowcomp avg(<field>) [by <field> [,<field>,...]] [sort {asc|desc} <field1> [, {asc|desc} <field2>,...]] [between 0|null|<number>|-<number> [and 0|null|<number>|-<number>] [frame_type=range]] [as <alias>]
```

### Description

The **avg()** function is used to return the average value of an integer field over a group of rows. The function syntax and application is based on the preceding stage:

#### comp stage

When the **avg** aggregation function is used with a comp stage, the function returns a single average value of an integer field for a group of rows, for all records that contain matching values for the fields identified in the **by** clause.

In addition, you can configure whether the raw data events are displayed by setting **addraddata** to either **true** or **false** (default), which are used to configure the final **comp** results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

#### windowcomp stage

When the **avg** aggregate function is used with a windowcomp stage, the function returns a single average value of an integer field for each row in the group of rows, for all records that contain matching values for the fields identified using a combination of the **by** clause, **sort**, and **between** window frame clause. The results are provided in a new column in the results table.

### Examples

#### comp example

Return a single average value of the **action\_total\_download** field for a group of rows, for all records that have matching values for their **actor\_process\_image\_path** and **actor\_process\_command\_line** values. The query calculates a maximum of 100 **xdr\_data** records and includes a **raw\_data** column listing a single value for the results.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp avg(Download) as average_download by Process_Path, Process_CMD
addraddata = true as raw_data
```

#### windowcomp example

Return the events that are above average per **Process\_Path** and **Process\_CMD**. The query returns a maximum of 100 **xdr\_data** records in a column called **avg\_download**.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| windowcomp avg(Download) by Process_Path, Process_CMD as avg_download
| filter Download > avg_download
```

## 6.4.19 | coalesce

### Abstract

Learn more about the Cortex Query Language **coalesce()** function that returns the first value that is not null from a defined list of fields.

### Syntax

```
coalesce (<field_1>, <field_2>,...<field_n>)
```

### Description

The **coalesce()** function takes an arbitrary number of arguments and returns the first value that is not NULL.



## Example

Given a list of fields that contain usernames, select the first one that is not `null` and display it in the `username` column.

```
dataset = xdr_data
| fields actor_primary_username,
  os_actor_primary_username,
  causality_actor_primary_username
| alter username = coalesce(actor_primary_username,
                           os_actor_primary_username,
                           causality_actor_primary_username)
```

## 6.4.20 | concat

### Abstract

Learn more about the Cortex Query Language `concat()` function joins multiple strings into a single string.

### Syntax

```
concat (<string1>, <string2>, ...)
```

### Description

The `concat()` function joins multiple strings into a single string. When using the `concat()` function with multiple fields and any of the fields have a null/empty value, the function returns empty.

### Example

Display the first non-NULL `action_boot_time` field value. In a second column called `abt_string`, use the `concat()` function to prepend "str: " to the value, and then display it.

```
dataset = xdr_data
| fields action_boot_time as abt
| filter abt != null
| alter abt_string = concat("str: ", to_string(abt))
| limit 1
```

## 6.4.21 | convert\_from\_base\_64

### Abstract

Learn more about the Cortex Query Language `convert_from_base_64` function.

### Syntax

```
convert_from_base_64("<base64-encoded input>")
```

### Description

The `convert_from_base_64()` function converts the base64-encoded input to the decoded string format.

### Example

Returns the decoded string format `Hello world` from the base64-encoded input "`SGVsbG8gd29ybGQ=`".

```
convert_from_base_64("SGVsbG8gd29ybGQ=")
```

## 6.4.22 | count

### Abstract

Learn more about the Cortex Query Language `count` function used with both `comp` and `windowcomp` stages.

### Syntax

#### comp stage

```
comp count([<field>]) [as <alias>] by <field_1>,<field_2> [addrawdata = true|false [as <target field>]]
```

#### windowcomp stage

```
windowcomp count([<field>]) [by <field> [,<field>,...]] [sort [asc|desc] <field1> [, [asc|desc] <field2>,...]] [between 0|null|<number>|-<number> [and 0|null|<number>|-<number>] [frame_type=range]] [as <alias>]
```



## Description

The `count()` function is used to return a single count for the number of rows either for a field over a group of rows, where only the number of non-null values found are returned, or without a field to count the number of rows, including null values. The function syntax and application is based on the preceding stage: comp stage

When the `count` aggregation function is used with a comp stage, the function returns one of the following:

- With a field: Returns a single count for the number of non-null rows, for all records that contain matching values for the fields identified in the `by` clause.
- Without a field: Counts the number of rows and includes null values.

In addition, you can configure whether the raw data events are displayed by setting `addraddata` to either `true` or `false` (default), which are used to configure the final `comp` results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

Use `count_distinct` to retrieve the number of unique values in the result set.

## windowcomp stage

When the `count` aggregate function is used with a windowcomp stage, the function returns one of the following:

- With a field: Returns a single count for the number of non-null rows for all records that contain matching values for the fields identified using a combination of the `by` clause, `sort`, and `between` window frame clause. The results are provided in a new column in the results table.
- Without a field: Counts the number of rows and includes null values.

## Examples

### comp example

Return a single count of all values found for the `actor_process_image_path` field in the group of rows, for all records that have matching values for their `actor_process_image_path` and `actor_process_command_line` values. The query calculates a maximum of 100 `xdr_data` records and includes a `raw_data` column listing a single value for the results.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp count(Process_Path) as num_process_path by process_path, process_cmd addrawdata = true as raw_data
| sort desc process_path
```

### windowcomp example

Return a single count for the number of values found in the `dns_query_name` field for each row in the group of rows, for all records that contain matching values in the `agent_ip_addresses` field. The query returns a maximum of 100 `xdr_data` records. The results are provided in the `count_dns_query_name` column.

```
dataset = xdr_data
| limit 100
| windowcomp count(dns_query_name) by agent_ip_addresses as count_dns_query_name
```

## 6.4.23 | `count_distinct`

### Abstract

Learn more about the Cortex Query Language `count_distinct` aggregate comp function that counts the number of unique values found for a field in the result set.

### Syntax

```
comp count_distinct(<field>) [as <alias>] by <field_1>,<field_2> [addraddata= true|false [as <target field>]]
```

## Description

The `count_distinct` aggregation is a comp function that returns a single value for the number of unique values found for a field over a group of rows, for all records that contain matching values for the fields identified in the `by` clause. This aggregate function is used in combination with a `comp` stage.

In addition, you can configure whether the raw data events are displayed by setting `addraddata` to either `true` or `false` (default), which are used to configure the final `comp` results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

Use `count` to retrieve the total number of values in the result set.

## Examples

Return a single count of the number of unique values found for the `actor_process_image_path` field over a group of rows, for all records that have matching values for their `actor_process_image_path` and `actor_process_command_line` values. The query calculates a maximum of 100



`xdr_data` records and includes a `raw_data` column listing the raw data events used to display the final `comp` results.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp count_distinct(Process_Path) as num_process_path by process_path, process_cmd addrawdata = true as raw_data
| sort desc process_path
```

## 6.4.24 | `current_time`

Abstract

Learn more about the Cortex Query Language `current_time()` function that returns the current time as a timestamp.

Syntax

```
current_time()
```

Description

The `current_time()` function returns a timestamp value representing the current time in the format `MMM dd YYYY HH:mm:ss`, such as `Jul 12th 2023 20:51:34`.

Example

From the `xdr_data` dataset, returns the events of the last 24 hours whose actor process started running more than 30 days ago.

```
dataset = xdr_data
| filter timestamp_diff(current_time(), to_timestamp(actor_process_execution_time, "MILLIS"), "DAY") > 30
```

## 6.4.25 | `date_floor`

Abstract

Learn more about the Cortex Query Language `date_floor()` function.

Syntax

```
date_floor (<timestamp field>, "<time unit>" [, "<time zone>"])
```

Description

The `date_floor()` function converts a timestamp value for a particular field or function result that contains a number, and returns a timestamp rounded down to the nearest whole value of a specified `<time unit>`, including a year (y), month (mo), week (w), day (d), or hour (h). The `<time zone>` offset is optional to configure using an hours offset, such as `+08:00`, or using a time zone name from the List of Supported Time Zones, such as "America/Chicago". When you do not configure a time zone, the default is UTC.

Example

Returns a maximum of 100 `xdr_data` records with the events of the `_time` field that are less than equal to a timestamp value. The timestamp value undergoes a number of different function manipulations. The current time is first rounded to the nearest whole value for the week according to the America/Los\_Angeles time zone. This timestamp value is then converted to the Unix epoch timestamp format in seconds and is added to the -2073600 Unix epoch time. This Unix epoch time value in seconds is then converted to the final timestamp value that is used to filter the `_time` fields and return the resulting records.

```
dataset = xdr_data
| filter _time < to_timestamp(add(to_epoch(date_floor(current_time(), "w", "America/Los_Angeles")), -2073600))
| limit 100
```

## 6.4.26 | `divide`

Abstract

Learn more about the Cortex Query Language `divide()` function that divides two integers.

Syntax

```
divide (<string> | <integer>, <string> | <integer>)
```



## Description

The `divide()` function divides two positive integers. Parameters can be either integer literals, or integers as a string type, such as might be contained in a data field.

## Example

```
dataset = xdr_data
| alter mynum = divide(action_file_size, 3)
| fields action_file_size, mynum
| filter action_file_size > 3
| limit 1
```

## 6.4.27 | earliest

### Abstract

Learn more about the Cortex Query Language `earliest` aggregate comp function that returns the earliest field value found with the matching criteria.

### Syntax

```
comp earliest(<field>) [as <alias>] by <field_1>,<field_2> [addrwdata = true|false [as <target field>]]
```

## Description

The `earliest` aggregation is a comp function that returns the chronologically earliest value found for a field over a group of rows that has matching values for the fields identified in the `by` clause. This function is dependent on a time-related field, so for your query to be considered valid, ensure that the dataset running this query contains a time-related field. This function is used in combination with a `comp` stage.

In addition, you can configure whether the raw data events are displayed by setting `addrwdata` to either `true` or `false` (default), which are used to configure the final `comp` results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

### Examples

Return the chronologically earliest timestamp found for any given `action_total_download` value for all records that have matching values for their `actor_process_image_path` and `actor_process_command_line` fields. The query calculates a maximum of 100 `xdr_data` records and includes a `raw_data` column listing the raw data events used to display the final `comp` results.

```
dataset = xdr_data
| fields _time, actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp earliest(_time) as download_time by Process_Path, Process_CMD addrwdata = true as raw_data
```

## 6.4.28 | extract\_time

### Abstract

Learn more about the Cortex Query Language `extract_time()` function that returns a specified portion of a timestamp.

### Syntax

```
extract_time (<timestamp>, <part>)
```

## Description

### IMPORTANT:

The `extract_time` values are based on the GMT time, even if you've adjusted the Timezone or Timestamp Format server settings as these configurations only affect how to display in Cortex AgentX. For more information on the server settings, see [Configure server settings](#).

The `extract_time()` function returns a specified part of a timestamp. The `part` parameter must be one of the following keywords:



- DAY
- DAYOFWEEK
- DAYOFEYEAR
- HOUR
- MICROSECOND
- MILLISECOND
- MINUTE
- MONTH
- QUARTER
- SECOND
- YEAR

#### Example

```
dataset = xdr_data
| alter timepart = extract_time(current_time(), "HOUR")
| fields timepart
| limit 1
```

### 6.4.29 | extract\_url\_host

#### Abstract

Learn more about the Cortex Query Language `extract_url_host()` function.

#### Syntax

```
extract_url_host ("<URL>")
```

#### Description

The `extract_url_host()` function returns the host of the URL. The function always returns a value in lowercase characters even if the URL provided contains uppercase characters.

#### Example

Output examples when using the function

Returns `paloaltonetworks.com` from the complete URL: `https://www.paloaltonetworks.com`.

```
extract_url_host ("https://www.paloaltonetworks.com")
```

Returns `a.b` for the URL: `//user:password@a.b:80/path?query`

```
extract_url_host ("//user:password@a.b:80/path?query")
```

Returns `www.example.co.uk` in lowercase for the complete URL: `www.Example.Co.UK`, which includes uppercase characters.

```
extract_url_host ("www.Example.Co.UK")
```

Returns `www.test.paloaltonetworks.com` for the following URL containing suffixes:

```
https://www.test.paloaltonetworks.com/suffix/another_suffix
```

```
extract_url_host ("https://www.test.paloaltonetworks.com/suffix/another_suffix")
```

#### Complete XQL Query Example

Returns one `xdr_data` record in the results table where the host of the URL `https://www.test.paloaltonetworks.com` is listed in the `URL_HOST` column as `www.test.paloaltonetworks.com`.

```
dataset = xdr_data
| alter url_host = extract_url_host("https://www.test.paloaltonetworks.com")
| fields url_host
| limit 1
```



## 6.4.30 | extract\_url\_pub\_suffix

### Abstract

Learn more about the Cortex Query Language `extract_url_pub_suffix()` function.

### Syntax

```
extract_url_pub_suffix ("<URL>")
```

### Description

The `extract_url_pub_suffix()` function returns the public suffix of the URL, such as com, org, or net. The function always returns a value in lowercase characters even if the URL provided contains uppercase characters.

### Example

Output examples when using the function

Returns com for the following URL: <https://paloaltonetworks.com>

```
extract_url_pub_suffix ("https://paloaltonetworks.com")
```

Returns com for the following URL containing suffixes: [https://www.test.paloaltonetworks.com/suffix/another\\_suffix](https://www.test.paloaltonetworks.com/suffix/another_suffix)

```
extract_url_pub_suffix ("https://www.test.paloaltonetworks.com/suffix/another_suffix")
```

### Complete XQL Query Example

Returns one `xdr_data` record in the results table where the public suffix of the URL <https://www.paloaltonetworks.com> is listed in the `URL_PUB_SUFFIX` column as com.

```
dataset = xdrl_data  
| alter url_pub_suffix = extract_url_pub_suffix("https://paloaltonetworks.com")  
| fields url_pub_suffix  
| limit 1
```

## 6.4.31 | extract\_url\_registered\_domain

### Abstract

Learn more about the Cortex Query Language `extract_url_registered_domain()` function.

### Syntax

```
extract_url_registered_domain ("<URL>")
```

### Description

The `extract_url_registered_domain()` function returns the registered domain or registerable domain, the public suffix plus one preceding label, of a URL. The function always returns a value in lowercase characters even if the URL provided contains uppercase characters.

### Examples

Output examples when using the function

Returns [paloaltonetworks.com](https://www.paloaltonetworks.com) from the complete URL: <https://www.paloaltonetworks.com>.

```
extract_url_registered_domain ("https://www.paloaltonetworks.com")
```

Returns NULL for the URL: //user:password@a.b:80/path?query

```
extract_url_registered_domain ("//user:password@a.b:80/path?query")
```

Returns example.co.uk in lowercase for the complete URL: [www.Example.Co.UK](http://www.Example.Co.UK), which includes uppercase characters.

```
extract_url_registered_domain ("www.Example.Co.UK")
```

Returns [paloaltonetworks.com](https://www.paloaltonetworks.com) for the following URL containing suffixes: [https://www.test.paloaltonetworks.com/suffix/another\\_suffix](https://www.test.paloaltonetworks.com/suffix/another_suffix)

```
extract_url_registered_domain ("https://www.test.paloaltonetworks.com/suffix/another_suffix")
```

### Complete XQL query example

Returns one `xdr_data` record in the results table where the registered domain of the URL <https://www.test.paloaltonetworks.com> is listed in the `REGISTERED_DOMAIN` column as [paloaltonetworks.com](https://www.paloaltonetworks.com).



```
dataset = xdr_data
| alter registered_domain = extract_url_registered_domain("https://www.test.paloaltonetworks.com")
| fields registered_domain
| limit 1
```

#### 6.4.32 | first

##### Abstract

Learn more about the Cortex Query Language **first** aggregate comp function that returns the first field value found in the dataset with the matching criteria.

##### Syntax

```
comp first(<field>) [as <alias>] by <field_1>,<field_2> [addrawdata = true|false [as <target field>]]
```

##### Description

The **first** aggregation is a comp function that returns a single first value found for a field in the dataset over a group of rows that has matching values for the fields identified in the **by** clause. This function is dependent on a time-related field, so for your query to be considered valid, ensure that the dataset running this query contains a time-related field. This function is used in combination with a **comp** stage.

In addition, you can configure whether the raw data events are displayed by setting **addrawdata** to either **true** or **false** (default), which are used to configure the final **comp** results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

##### Examples

Return the first timestamp found in the dataset for any given **action\_total\_download** value for all records that have matching values for their **actor\_process\_image\_path** and **actor\_process\_command\_line** fields. The query calculates a maximum of 100 **xdr\_data** records and includes a **raw\_data** column listing the raw data events used to display the final **comp** results.

```
dataset = xdr_data
| fields _time,actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp first(_time) as download_time by Process_Path, Process_CMD addrawdata = true as raw_data
```

#### 6.4.33 | first\_value

##### Abstract

Learn more about the Cortex Query Language **first\_value()** navigation function that is used with a **windowcomp** stage.

##### Syntax

```
windowcomp first_value(<field>) [by <field> [,<field>,...]] sort [asc|desc] <field1> [, [asc|desc] <field2>,...] [between 0|null|<number>|-<number> [and 0|null|<number>|-<number>] [frame_type=range]] [as <alias>]
```

##### Description

The **first\_value()** function is a navigation function that is used in combination with a **windowcomp** stage. This function is used to return a single value of a field for the first row of each row in the group of rows in the current window frame, for all records that contain matching values for the fields identified using a combination of the **by** clause, **sort** (mandatory), and **between** window frame clause.

##### Example

Return the first IP address a user authenticated from successfully.

```
preset = authentication_story
| filter auth_identity not in (null, "") and auth_outcome = """SUCCESS"" and action_country != UNKNOWN
| alter et = to_epoch(_time), t = _time
| bin t span = 1d
| limit 100
| windowcomp first_value(action_local_ip) by auth_identity, t sort asc et between null and null as first_action_local_ip
| fields auth_identity , *action_local_ip
```

#### 6.4.34 | floor

##### Abstract

Learn more about the Cortex Query Language **floor()** function that rounds a field that contains a number down to the nearest whole integer.

##### Syntax

```
floor (<number>)
```



## Description

The `floor()` function converts a field that contains a number, and returns an integer rounded down to the nearest whole number.

### 6.4.35 | `format_string`

#### Abstract

Learn more about the Cortex Query Language `format_string()` function.

#### Syntax

```
format_string("<format string>", <field_1>, <field_2>,...<field_n> )
```

#### Description

The `format_string()` function returns a string from a format string that contains zero or more format specifiers, along with a variable length list of additional arguments that matches the format specifiers. A format specifier is initiated by the % symbol, and must map to one or more of the remaining arguments. Usually, this is a one-to-one mapping, except when the \* specifier is used.

#### Examples

- STRING

```
dataset = xdr_data
| alter styled_action_category_appID = format_string("-%s-", action_category_of_app_id )
| fields styled_action_category_appID
| limit 100
```

- Simple integer

```
dataset = xdr_data
| filter action_remote_ip_int != null
| alter simple_int = format_string("%d", action_remote_ip_int)
| fields simple_int
| limit 100
```

- Integer with left blank padding

```
dataset = xdr_data
| filter action_remote_ip_int != null
| alter int_with_left_blank = format_string("|%100d|", action_remote_ip_int)
| fields int_with_left_blank
| limit 100
```

- Integer with left zero padding

```
dataset = xdr_data
| filter action_remote_ip_int != null
| alter int_with_left_zero_padding = format_string(">%0100d+", action_remote_ip_int)
| fields int_with_left_zero_padding
| limit 100
```

### 6.4.36 | `format_timestamp`

#### Abstract

Learn more about the Cortex Query Language `format_timestamp()` function that returns a string after formatting a timestamp according to a specified string format.

#### Syntax

```
format_timestamp("<format string>", <timestamp field>)
format_timestamp("<format string>", <timestamp field>, "<time zone>")
```

#### Description

The `format_timestamp()` function returns a string after formatting a timestamp according to a specified string format. The `<time zone>` is optional to configure using an hours offset, such as `+08:00`, or using a time zone name from the List of Supported Time Zones, such as "America/Chicago". The `format_timestamp()` function should include an alter stage. For more information, see the examples below.

#### Examples



- Without a time zone configured

Returns a maximum of 100 `xdr_data` records, which includes a string field called `new_time` in the format `YYYY/MM/dd HH:mm:ss`, such as `2021/11/12 12:10:30`. This format is detailed in the `format_timestamp` function, which defines retrieving the `new_time` (`%Y/%m/%d %H:%M:%S`) from the `_time` field.

```
dataset = xdr_data
| alter new_time = format_timestamp("%Y/%m/%d %H:%M:%S", _time)
| fields new_time
| limit 100
```

- With a time zone configured using an hours offset

Returns a maximum of 100 `xdr_data` records, which includes a string field called `new_time` in the format `YYYY/MM/dd HH:mm:ss`, such as `2021/11/12 01:53:35`. This format is detailed in the `format_timestamp` function, which defines the retrieving the `new_time` (`%Y/%m/%d %H:%M:%S`) from the `_time` field and adding `+03:00` hours as the time zone format.

```
dataset = xdr_data
| alter new_time = format_timestamp("%Y/%m/%d %H:%M:%S", _time, "+03:00")
| fields new_time
| limit 100
```

- With a time zone name configured

Returns a maximum of 100 `xdr_data` records, which includes a string field called `new_time` in the format `YYYY/MM/dd HH:mm:ss`, such as `2021/11/12 01:53:35`. This format is detailed in the `format_timestamp` function, which defines the retrieving the `new_time` (`%Y/%m/%d %H:%M:%S`) from the `_time` field, and includes an "America/Chicago" time zone.

```
dataset = xdr_data
| fields _time
| alter new_time = format_timestamp("%Y/%m/%d %H:%M:%S", _time, "America/Chicago")
| fields new_time
| limit 100
```

## 6.4.37 | if

### Abstract

Learn more about the Cortex Query Language `if()` function that returns a result after evaluating a condition.

### Syntax

Regular if statement

```
if (<boolean expression>, <true return expression>[, <false return expression>])
```

Nested if/else statement

- `if(<boolean expression1>, <true return expression1>, <boolean expression2>, <true return expression2>[, <boolean expression3>, <true return expression3>,...][, <false return expression>])`
- `if(<boolean expression1>, if(<boolean expression2>, <true return expression2> [,<false return expression2>])...[,<false return expression1>])`

### NOTE:

In the above syntax, `if(<boolean expression2>, <true return expression2> [,<false return expression2>])` represents the `<true return expression1>`.

### Description

The `if()` function evaluates a single expression or group of expressions depending on the syntax used to define the function. The syntax can be set up in the following ways:

- Regular if statement: A single boolean expression is evaluated. If the expression evaluates as `true`, the function returns the results defined in the second function argument. If the expression evaluates as `false` and a false return expression is defined, the function returns the results of the third function argument; otherwise, if no false return expression is set, returns null.
- Nested if/else statement: At least two boolean expressions and two true return expressions are required when using this option. The first boolean expression is evaluated. If the first expression evaluates as `true`, the function returns the results defined in the second function argument. The second boolean expression is evaluated. If the second expression evaluates as `true`, the function returns the results defined in the fourth function argument. If there are any other boolean expressions defined, they are evaluated following the same pattern when evaluated as `true`. If any of the expressions evaluates as `false` and a false return expression is defined, the function returns the results defined in the last function argument for the false return expression; otherwise, if no false return expression is set, returns null.

### Examples

Regular if statement



If '.exe' is present on the `action_process_image_name` field value, replace that substring with an empty string. This example uses the `replace` and `lowercase` functions, as well as the `contains` operator to perform the conditional check. When the '.exe' is not present, the value is returned as is.

```
dataset = xdr_data
| fields action_process_image_name as apin
| filter apin != null
| alter remove_exe_process =
  if(lowercase(apin) contains ".exe", // boolean expression
    replace(lowercase(apin),".exe","", // return if true
    lowercase(apin)) // return if false
| limit 10
Nested if/else statement
```

Return a maximum of 1 `xdr_data` record from the past 7 days. The table results include a new column called `check_ip`, which evaluates and returns the following:

- If the `action_local_ip` contains an IP address that begins with 10, return `Local 10`.
- If the `action_local_ip` contains an IP address that begins with 172, return `Local 172 ?`.
- If the `action_local_ip` contains an IP address that begins with 192.168, return `Local 192`.
- If all the above expressions evaluate as `false`, return null.

```
config timeframe = 7d | dataset = xdr_data
| limit 1
| alter
  check_ip = if(action_local_ip == "^10",//boolean expression1
  "Local 10", // true return expression1
  action_local_ip == "172", //boolean expression2
  "Local 172 ?", //true return expression2
  action_local_ip == "192\168", //boolean expression3
  "Local 192") //true return expression3
```

## 6.4.38 | incidr

### Abstract

Learn more about the Cortex Query Language `incidr()` function.

### Syntax

```
incidr(<IPv4_address>, <CIDR1_range1> | <CIDR1_range1, CIDR2_range2, ...>)
```

### Description

The `incidr()` function accepts an IPv4 address, and an IPv4 range or comma separated IPv4 ranges using CIDR notation, and returns `true` if the address is in range. Both the IPv4 address and CIDR ranges can be either an explicit string using quotes (" "), such as "192.168.0.1", or a string field.

#### NOTE:

The first parameter must contain an IPv4 address contained in an IPv4 field. For production purposes, this IPv4 address will normally be carried in a field that you retrieve from a dataset. For manual usage, assign the IPv4 address to a field, and then use that field with this function.

Multiple CIDRs are defined with comma separated syntax when building an XQL query with the Query Builder or in Correlation Rules. When defining multiple CIDRs, the logical OR is used between the CIDRS listed, so as long as one address is in range the entire statement returns `true`. Here are a few examples of how this logic works to determine whether the `incidr()` function returns `true` and displays results or `false`, where no results are displayed:

- Function returns `true` and results are displayed:

```
dataset = test
| alter ip_address = "192.168.0.1"
| filter incidr(ip_address, "192.168.0.0/24, 1.168.0.0/24") = true
```

- Function returns `false` and no results are displayed:

```
dataset = test
| alter ip_address = "192.168.0.1"
| filter incidr(ip_address, "2.168.0.0/24, 1.168.0.0/24") = true
```

- Function returns `false` and no results are displayed:

```
dataset = test
| alter ip_address = "192.168.0.1"
| filter incidr(ip_address, "192.168.0.0/24, 1.168.0.0/24") = false
```

- Function returns `true` and results are displayed:

```
dataset = test
| alter ip_address = "192.168.0.1"
| filter incidr(ip_address, "2.168.0.0/24, 1.168.0.0/24") = false
```



#### **NOTE:**

The same logic is used when using the `incidr` and `not incidr` operators. For more information, see [Supported operators](#).

#### Examples

Return a maximum of 10 `xdr_data` records, if the IPv4 address (192.168.10.14) is in range by verifying against a single CIDR (192.168.10.0/24):

```
alter my_ip = "192.168.10.14"
| alter inrange = incidr(my_ip, "192.168.10.0/24")
| fields inrange
| limit 10
```

Return a maximum of 10 `xdr_data` records, if the IPv4 address (192.168.0.1) is in range by verifying against multiple CIDRs (192.168.0.0/24 or 1.168.0.0/24):

```
dataset = xdr_data
| alter ip_address = "192.168.0.1"
| filter incidr(ip_address, "192.168.0.0/24, 1.168.0.0/24") = true
| limit 10
```

### 6.4.39 | `incidr6`

#### Abstract

Learn more about the Cortex Query Language `incidr6()` function.

#### Syntax

```
incidr6(<IPv6_address>, <CIDR1_range1> | <CIDR1_range1, CIDR2_range2, ...>)
```

#### Description

The `incidr6()` function accepts an IPv6 address, and an IPv6 range or comma separated IPv6 ranges using CIDR notation, and returns `true` if the address is in range. Both the IPv6 address and CIDR ranges can be either an explicit string using quotes (" "), such as

"3031:3233:3435:3637:3839:4041:4243:4445", or a string field.

#### **NOTE:**

The first parameter must contain an IPv6 address contained in an IPv6 field. For production purposes, this IPv6 address will normally be carried in a field that you retrieve from a dataset. For manual usage, assign the IPv6 address to a field, and then use that field with this function.

Multiple CIDRs are defined with comma separated syntax when building an XQL query with the Query Builder or in Correlation Rules. When defining multiple CIDRs, the logical OR is used between the CIDRS listed, so as long as one address is in range the entire statement returns `true`. Here are a few examples of how this logic works to determine whether the `incidr6()` function returns `true` and displays results or `false`, where no results are displayed:

- Function returns `true` and results are displayed:

```
dataset = test
| alter ip_address = "3031:3233:3435:3637:3839:4041:4243:4445"
| filter incidr(ip_address, "3031:3233:3435:3637:0000:0000:0000:0000/64, 6081:6233:6435:6637:0000:0000:0000:0000/64") = true
```

- Function returns `false` and no results are displayed:

```
dataset = test
| alter ip_address = "3031:3233:3435:3637:3839:4041:4243:4445"
| filter incidr(ip_address, "6081:6233:6435:6637:0000:0000:0000:0000/64, 7081:7234:7435:7737:0000:0000:0000:0000/64, fe80::/10") = true
```

- Function returns `false` and no results are displayed:

```
dataset = test
| alter ip_address = "3031:3233:3435:3637:3839:4041:4243:4445"
| filter incidr(ip_address, "3031:3233:3435:3637:0000:0000:0000:0000/64, 7081:7234:7435:7737:0000:0000:0000:0000/64, fe80::/10") = false
```

- Function returns `true` and results are displayed:

```
dataset = test
| alter ip_address = "3031:3233:3435:3637:3839:4041:4243:4445"
| filter incidr(ip_address, "6081:6233:6435:6637:0000:0000:0000:0000/64, 7081:7234:7435:7737:0000:0000:0000:0000/64, fe80::/10") = false
```

#### **NOTE:**

The same logic is used when using the `incidr6` and `not incidr6` operators. For more information, see [Supported operators](#).

#### Example

Return a maximum of 10 `xdr_data` records, if the IPv6 address (3031:3233:3435:3637:3839:4041:4243:4445) is in range by verifying against a single CIDR (3031:3233:3435:3637:0000:0000:0000:0000/64):

```
alter my_ip = "3031:3233:3435:3637:3839:4041:4243:4445"
| alter inrange = incidr6(my_ip, "3031:3233:3435:3637:0000:0000:0000:0000/64")
```



```
| fields inrange  
| limit 10
```

Return a maximum of 10 `xdr_data` records, if the IPv6 address (3031:3233:3435:3637:3839:4041:4243:4445) is in range by verifying against multiple CIDRs (2001:0db8:85a3:0000:0000:8a2e:0000:0000/64 or fe80::/10):

```
dataset = xdru_data  
| alter ip_address = "fe80::1"  
| filter incidr6(ip_address, "2001:0db8:85a3:0000:0000:8a2e:0000:0000/64, fe80::/10") = true  
| limit 10
```

#### 6.4.40 | `incidrlist`

##### Abstract

Learn more about the Cortex Query Language `incidrlist()` function.

##### Syntax

```
incidrlist(<IP_address list>, <CIDR_range>)
```

##### Description

The `incidrlist()` function accepts a string containing a comma-separated list of IP addresses, and an IP range using CIDR notation, and returns `true` if all the addresses are in range.

##### Examples

Return `true` if the list of IP addresses fall within the specified IP range. Note that the input type is a comma-separated list of IP addresses, and not an array of IP addresses.

```
alter inrange = incidrlist("192.168.10.16,192.168.10.3",  
                           "192.168.10.0/24")  
| fields inrange  
| limit 1
```

If you want to evaluate a true array of IP addresses, convert the array to a comma-separated list using `arraystring()`. For example, using the `pan_ngfw_traffic_raw` dataset:

```
dataset = panw_ngfw_traffic_raw  
| filter dest_ip != null  
| comp values(dest_ip) as dips by source_ip,action  
| alter dips = arraystring(dips, ", ")  
| alter inrange = incidrlist(dips, "192.168.10.0/24")  
| fields source_ip, action, dips, inrange  
| limit 100
```

#### 6.4.41 | `int_to_ip`

##### Abstract

Learn more about the Cortex Query Language `int_to_ip()` function that safely converts a signed integer representation of an IPv4 address to a string equivalent.

##### Syntax

```
int_to_ip(<IPv4_integer>)
```

##### Description

The `int_to_ip()` function tries to safely convert a signed integer representation of an IPv4 address into its string equivalent.

##### Examples

Returns the IPv4 address "4.130.58.140" from the integer representation of the IPv4 address provided as 75643532.

```
int_to_ip(75643532)
```

Returns the IPv4 address "251.125.197.116" from the integer representation of the IPv4 address provided as -75643532.

```
int_to_ip(-75643532)
```



## 6.4.42 | ip\_to\_int

### Abstract

Learn more about the Cortex Query Language `ip_to_int()` function that safely converts a string representation of an IPv4 address to an integer equivalent.

### Syntax

```
ip_to_int(<IPv4_address>)
```

#### NOTE:

This function was previously called `safe_ip_to_int()` and was renamed to `ip_to_int()`.

### Description

The `ip_to_int()` function tries to safely convert a string representation of an IPv4 address into its integer equivalent.

### Example

Returns the integer 808530483 from the string representation of the IPv4 address provided as "48.49.50.51".

```
ip_to_int("48.49.50.51")
```

## 6.4.43 | is\_ipv4

### Abstract

Learn more about the Cortex Query Language `is_ipv4()` function.

### Syntax

```
is_ipv4(<IPv4_address>)
```

### Description

The `is_ipv4()` function accepts a string, and returns `true` if the string is a valid IPv4 address. The IPv4 address can be either an explicit string using quotes (" "), such as "192.168.0.1", or a string field.

#### NOTE:

The `<IPv4_address>` must contain an IPv4 address in an IPv4 field. For production purposes, this IPv4 address will normally be carried in a field that you retrieve from a dataset. For manual usage, assign the IPv4 address to a field, and then use that field with this function.

### Example

Data table for `ips_test_raw` dataset

The example provided is based on the following data table for a dataset called `ips_test_raw`:

_TIME	IP	_VENDOR	_PRODUCT
Mar 26th 2025 19:26:07	1.1.1.1	ips	test
Mar 26th 2025 19:26:07	192.168.1.100	ips	test
Mar 26th 2025 19:26:07	FF0E::1	ips	test
Mar 26th 2025 19:26:07	127.0.0.1	ips	test
Mar 26th 2025 19:26:07	172.32.0.1	ips	test
Mar 26th 2025 19:26:07	2606:4700:4700::1111	ips	test

Query: Filter the IPv4 addresses



```

dataset = ips_test_raw
| alter IsIpv4 = is_ipv4(ip)
| filter IsIpv4
Output results table

```

Returns all the IPv4 addresses from the `ip` field in the `ips_test_raw` dataset. When the `is_ipv4` function returns `true`, the results are displayed with a new `IsIpv4` column (field) indicating a true value. If the function returns `false`, no results are returned.

<code>_TIME</code>	<code>IP</code>	<code>_VENDOR</code>	<code>_PRODUCT</code>	<code>ISIPV4</code>
Mar 26th 2025 19:26:07	1.1.1.1	ips	test	true
Mar 26th 2025 19:26:07	192.168.1.100	ips	test	true
Mar 26th 2025 19:26:07	127.0.0.1	ips	test	true
Mar 26th 2025 19:26:07	172.32.0.1	ips	test	true

#### 6.4.44 | `is_known_private_ipv4`

##### Abstract

Learn more about the Cortex Query Language `is_known_private_ipv4()` function.

##### Syntax

```
is_known_private_ipv4(<IPv4_address>)
```

##### Description

The `is_known_private_ipv4()` function accepts an IPv4 address, and returns `true` if the IPv4 string address belongs to any of the following known set of private network IPs:

- 10.0.0.0/8
- 172.16.0.0/12
- 192.168.0.0/16

The IPv4 address can be either an explicit string using quotes (" "), such as "192.168.0.1", or a string field.

##### NOTE:

The `<IPv4_address>` must contain an IPv4 address in an IPv4 field. For production purposes, this IPv4 address will normally be carried in a field that you retrieve from a dataset. For manual usage, assign the IPv4 address to a field, and then use that field with this function.

##### Example

Data table for `ips_test_raw` dataset

The example provided is based on the following data table for a dataset called `ips_test_raw`:

<code>_TIME</code>	<code>IP</code>	<code>_VENDOR</code>	<code>_PRODUCT</code>
Mar 26th 2025 19:26:07	1.1.1.1	ips	test
Mar 26th 2025 19:26:07	192.168.1.100	ips	test
Mar 26th 2025 19:26:07	FF0E::1	ips	test



<u>TIME</u>	<u>IP</u>	<u>VENDOR</u>	<u>PRODUCT</u>
Mar 26th 2025 19:26:07	127.0.0.1	ips	test
Mar 26th 2025 19:26:07	172.32.0.1	ips	test
Mar 26th 2025 19:26:07	2606:4700:4700::1111	ips	test

Query: Filter the IPv4 addresses belonging to a set of known private network IPs

```
dataset = ips_test_raw
| alter IsKnownPrivateIpv4 = is_known_private_ipv4(ip)
| filter IsKnownPrivateIpv4
```

Output results table

Returns all the IPv4 addresses that belong to a set of known private network IPs from the `ip` field in the `ips_test_raw` dataset. When the `is_known_private_ipv4` function returns `true`, the results are displayed with a new `IsKnownPrivateIpv4` column (field) indicating a true value. If the function returns `false`, no results are returned.

<u>TIME</u>	<u>IP</u>	<u>VENDOR</u>	<u>PRODUCT</u>	<u>ISKNOWNRIVATEIPV4</u>
Mar 26th 2025 19:26:07	192.168.1.100	ips	test	true

#### 6.4.45 | `is_ipv6`

Abstract

Learn more about the Cortex Query Language `is_ipv6()` function.

Syntax

```
is_ipv6(<IPv6_address>)
```

Description

The `is_ipv6()` function accepts a string, and returns `true` if the string is a valid IPv6 address. The IPv6 address can be either an explicit string using quotes (" "), such as "3031:3233:3435:3637:3839:4041:4243:4445", or a string field.

#### NOTE:

The `<IPv6_address>` must contain an IPv6 address in an IPv6 field. For production purposes, this IPv6 address will normally be carried in a field that you retrieve from a dataset. For manual usage, assign the IPv6 address to a field, and then use that field with this function.

Example

Data table for `ips_test_raw` dataset

The example provided is based on the following data table for a dataset called `ips_test_raw`:

<u>TIME</u>	<u>IP</u>	<u>VENDOR</u>	<u>PRODUCT</u>
Mar 26th 2025 19:26:07	1.1.1.1	ips	test
Mar 26th 2025 19:26:07	192.168.1.100	ips	test
Mar 26th 2025 19:26:07	FF0E::1	ips	test



<u>TIME</u>	<u>IP</u>	<u>VENDOR</u>	<u>PRODUCT</u>
Mar 26th 2025 19:26:07	127.0.0.1	ips	test
Mar 26th 2025 19:26:07	172.32.0.1	ips	test
Mar 26th 2025 19:26:07	2606:4700:4700::1111	ips	test

Query: Filter the IPv6 addresses

```
dataset = ips_test_raw
| alter IsIpv6 = is_ipv6(ip)
| filter IsIpv6
```

Output results table

Returns all the IPv6 addresses from the `ip` field in the `ips_test_raw` dataset. When the `is_ipv6` function returns `true`, the results are displayed with a new `IsIpv6` column (field) indicating a true value. If the function returns `false`, no results are returned.

<u>TIME</u>	<u>IP</u>	<u>VENDOR</u>	<u>PRODUCT</u>	<u>ISIPV6</u>
Mar 26th 2025 19:26:07	FF0E::1	ips	test	true
Mar 26th 2025 19:26:07	2606:4700:4700::1111	ips	test	true

#### 6.4.46 | `is_known_private_ipv6`

Abstract

Learn more about the Cortex Query Language `is_known_private_ipv6()` function.

Syntax

```
is_known_private_ipv6(<IPv6_address>)
```

Description

The `is_known_private_ipv6()` function accepts an IPv6 address, and returns `true` if the IPv6 string address belongs to any of the following known set of private network IPs:

- FC00::/7
- FD00::/7

The IPv6 address can be either an explicit string using quotes (" "), such as "3031:3233:3435:3637:3839:4041:4243:4445", or a string field.

#### NOTE:

The `<IPv6_address>` must contain an IPv6 address in an IPv6 field. For production purposes, this IPv6 address will normally be carried in a field that you retrieve from a dataset. For manual usage, assign the IPv6 address to a field, and then use that field with this function.

Example

Data table for `ips_test_raw` dataset

The example provided is based on the following data table for a dataset called `ips_test_raw`:

<u>TIME</u>	<u>IP</u>	<u>VENDOR</u>	<u>PRODUCT</u>
Mar 26th 2025 19:26:07	FC00::1	ips	test



<u>TIME</u>	<u>IP</u>	<u>VENDOR</u>	<u>PRODUCT</u>
Mar 26th 2025 19:26:07	192.168.1.100	ips	test
Mar 26th 2025 19:26:07	FF0E::1	ips	test
Mar 26th 2025 19:26:07	127.0.0.1	ips	test
Mar 26th 2025 19:26:07	172.32.0.1	ips	test
Mar 26th 2025 19:26:07	2606:4700:4700::1111	ips	test

Query: Filter the IPv6 addresses belonging to a set of known private network IPs

```
dataset = ips_test_raw
| alter IsKnownPrivateIpv6 = is_known_private_ipv6(ip)
| filter IsKnownPrivateIpv6
```

Output results table

Returns all the IPv6 addresses that belong to a set of known private network IPs from the `ip` field in the `ips_test_raw` dataset. When the `is_known_private_ipv6` function returns `true`, the results are displayed with a new `IsKnownPrivateIpv6` column (field) indicating a true value. If the function returns `false`, no results are returned.

<u>TIME</u>	<u>IP</u>	<u>VENDOR</u>	<u>PRODUCT</u>	<u>ISKNOWNPRIVATEIPV6</u>
Mar 26th 2025 19:26:07	FC00::1	ips	test	true

#### 6.4.47 | json\_extract

Abstract

Learn more about the Cortex Query Language `json_extract()` function that accepts a string representing a JSON object, and returns a field value from that object.

##### **IMPORTANT:**

Before using this JSON function, it's important that you understand how Cortex AgentiX treats a JSON in the Cortex Query Language. For more information, see [JSON functions](#).

Syntax

Regular Syntax

```
json_extract(<json_object_formatted_string>, <json_path>)
```

When a field in the `<json_path>` contains characters, such as a dot (.) or colon (:), use the syntax:

```
json_extract(<json_object_formatted_string>, "[ '<json_field>' ]")
```

Syntactic Sugar Format

To make it easier for you to write your XQL queries, you can also use the following syntactic sugar format.

```
<json_object_formatted_string> -> <json_path>{}
```

When a field in the `<json_path>` contains characters, such as a dot (.) or colon (:), use the syntax:

```
<json_object_formatted_string> -> [ "<json_field>" ]{}
```

Description

The `json_extract()` function extracts inner JSON objects by retrieving the value from the identified field. The returned datatype is always a string. If the input string does not represent a JSON object, this function fails to parse. To convert a string field to a JSON object, use the `to_json_string` function.



#### **IMPORTANT:**

JSON field names are case sensitive, so the key to field pairing must be identical in an XQL query for results to be found. For example, if a field value is "TIMESTAMP" and your query is defined to look for "timestamp", no results will be found.

#### **NOTE:**

The field value is always returned as a string. To return the scalar values, which are not an object or an array, use `json_extract_scalar`.

##### Examples

Return the `storage_device_name` value from the `action_file_device_info` field.

```
dataset = xdru_data
| fields action_file_device_info as afdi
| alter sdn = json_extract(to_json_string(afdi), ".$storage_device_name")
| filter afdi != null
```

##### Using Syntactic Sugar Format

The same example above with a syntactic sugar format.

```
dataset = xdru_data
| fields action_file_device_info as afdi
| alter sdn = to_json_string(afdi)->storage_device_name{}
| filter afdi != null
```

#### 6.4.48 | `json_extract_array`

##### Abstract

Learn more about the Cortex Query Language `json_extract_array()` function that accepts a string representing a JSON array, and returns an XQL-native array.

#### **IMPORTANT:**

Before using this JSON function, it's important that you understand how Cortex AgentIX treats a JSON in the Cortex Query Language. For more information, see [JSON functions](#).

##### Syntax

###### Regular Syntax

```
json_extract_array(<json_array_string>, <json_path>)
```

When a field in the `<json_path>` contains characters, such as a dot (.) or colon (:), use the syntax:

```
json_extract_array(<json_array_string>, "[<json_field>]")
```

###### Syntactic Sugar Format

To make it easier for you to write your XQL queries, you can also use the following syntactic sugar format.

```
<json_array_string> -> <json_path>[]
```

When a field in the `<json_path>` contains characters, such as a dot (.) or colon (:), use the syntax:

```
<json_array_string> -> [<json_field>][]
```

##### Description

The `json_extract_array()` function accepts a string representing a JSON array, and returns an XQL-native array. To convert a string field to a JSON object, use the `to_json_string` function.

#### **IMPORTANT:**

JSON field names are case sensitive, so the key to field pairing must be identical in an XQL query for results to be found. For example, if a field value is "TIMESTAMP" and your query is defined to look for "timestamp", no results will be found.

##### Examples

###### Regular Syntax

Extract the first IPV4 address found in the first element of the `agent_interface_map` array.

```
dataset = xdru_data
| fields agent_interface_map as aim
| alter ipv4 = json_extract_array(to_json_string(arrayindex(aim, 0)), ".$ipv4")
| filter aim != null
| limit 10
```



## Syntactic Sugar Format

The same example above with a syntactic sugar format.

```
dataset = xdr_data
| fields agent_interface_map as aim
| alter ipv4 = to_json_string(aim)->[0].ipv4[0]
| filter aim != null
| limit 10
```

### 6.4.49 | json\_extract\_scalar

#### Abstract

Learn more about the Cortex Query Language `json_extract_scalar()` function.

#### **IMPORTANT:**

Before using this JSON function, it's important that you understand how Cortex AgentX treats a JSON in the Cortex Query Language. For more information, see [JSON functions](#).

#### Syntax

##### Regular Syntax

```
json_extract_scalar(<json_object_formatted_string>, <field_path>)
```

When a field in the `<json_path>` contains characters, such as a dot (.) or colon (:), use the syntax:

```
json_extract_scalar(<json_object_formatted_string>, "[<json_field> ]")
```

##### Syntactic Sugar Format

To make it easier for you to write your XQL queries, you can also use the following syntactic sugar format:

```
<json_object_formatted_string> -> <field_path>
```

When a field in the `<json_path>` contains characters, such as a dot (.) or colon (:), use the syntax:

```
<json_object_formatted_string> -> ["<json_field>"]
```

#### Description

The `json_extract_scalar()` function accepts a string representing a JSON object, and it retrieves the value from the identified field as a string. This function always returns a string. If the JSON field is an object or array, it will return a null value. To retrieve an XQL-native datatype, use an appropriate function, such as `to_float` or `to_integer`. If the input string does not represent a JSON object, this function fails to parse. To convert a string field to a JSON object, use the `to_json_string` function.

#### **IMPORTANT:**

JSON field names are case sensitive, so the key to field pairing must be identical in an XQL query for results to be found. For example, if a field value is "TIMESTAMP" and your query is defined to look for "timestamp", no results will be found.

#### Examples

Return the `storage_device_drive_type` value from the `action_file_device_info` field, and return the record if it is 1.

There are two ways that you can build this query either with a filter using an XQL-native datatype or string.

Option A - Filter using an XQL-native datatype

```
dataset = xdr_data
| fields action_file_device_info as afdi
| alter sdn = to_integer(json_extract_scalar(to_json_string(afdi), ".$storage_device_drive_type"))
| filter sdn = 1
| limit 10
```

Option B - Filter using a string

```
dataset = xdr_data
| fields action_file_device_info as afdi
| alter sdn = json_extract_scalar(to_json_string(afdi), ".$storage_device_drive_type")
| filter sdn = "1"
| limit 10
```

#### Using Syntactic Sugar Format

The same example above with a syntactic sugar format.

```
dataset = xdr_data
| fields action_file_device_info as afdi
| alter sdn = to_integer(to_json_string(afdi)->storage_device_drive_type)
```



```
| filter sdn = 1  
| limit 10
```

## 6.4.50 | json\_extract\_scalar\_array

### Abstract

Learn more about the Cortex Query Language `json_extract_scalar_array()` function.

#### IMPORTANT:

Before using this JSON function, it's important that you understand how Cortex AgentX treats a JSON in the Cortex Query Language. This function doesn't have a syntactic sugar format. For more information, see [JSON functions](#).

### Syntax

```
json_extract_scalar_array(<json_array_string>, <json_path>)
```

#### IMPORTANT:

When a Cortex Data Model (XDM) field is used in the `<json_path>` and contains a dot (.) character, such as `xdm.source.host.device_id`, use the syntax:

```
json_extract_scalar_array(<json_array_string>, "[<json_field>'"]")
```

All other characters in the `<json_path>`, such as colon (:), should be escaped as it's an invalid JSON path, and are currently unsupported.

### Description

The `json_extract_scalar_array()` function accepts a string representing a JSON array, and returns an XQL-native array. This function is equivalent to the `json_extract_array` except that the final output isn't displayed in double quotes ("..."). To convert a string field to a JSON object, use the `to_json_string` function.

#### IMPORTANT:

JSON field names are case sensitive, so the key to field pairing must be identical in an XQL query for results to be found. For example, if a field value is "TIMESTAMP" and your query is defined to look for "timestamp", no results will be found.

### Examples

#### Dataset query example

Extract the first IPv4 address found in the first element of the `agent_interface_map` array. The values of the IPv4 addresses in the array will not contain any double quotes.

```
dataset = xdr_data  
| fields agent_interface_map as aim  
| alter ipv4 = json_extract_scalar_array(to_json_string(arrayindex(aim, 0)), "$.ipv4")  
| filter aim != null  
| limit 10
```

Final output with 1 row from the results table. Notice that the IPv4 column doesn't contain any double quotes (" ") around the IP address 172.16.15.42:

_TIME	AIM	_PRODUCT	_VENDOR	INSERT_TIMESTAMP	IPV4
Aug 9th 2023 10:04:39	[{"ipv4": ["172.16.15.42"], "ipv6": [], "mac": "00:50:56:9f:30:a9"}]	XDR agent	PANW	Aug 17th 2023 19:25:48	172.16.15.42

In contrast, compare the above results to the same query using the `json_extract_array()` function. The final output with the same row from the results table has in the IPv4 column the IP address in double quotes "172.16.15.42".

_TIME	AIM	_PRODUCT	_VENDOR	INSERT_TIMESTAMP	IPV4
Aug 9th 2023 10:04:39	[{"ipv4": ["172.16.15.42"], "ipv6": [], "mac": "00:50:56:9f:30:a9"}]	XDR agent	PANW	Aug 17th 2023 19:25:48	"172.16.15.42"

#### XDM example

Extract the `xdm.file.permissions.owner` in the `xdm.finding.normalized_fields` array, which returns true when the string value is one of the following: r or w.



```
dataset = findings
| filter JSON_EXTRACT_SCALAR_ARRAY(xdm.finding.normalized_fields, "$['xdm.file.permissions.owner'])") in ("r", "w")
| fields xdm.finding.normalized_fields
```

Output results:

_TIME	XDM.FINDING.NORMALIZED_FIELDS
Jan 15th 2025 09:10:44	{ "xdm.file.permissions.owner": [ "r", "w", "x" ] }

#### 6.4.51 | json\_path\_extract

Abstract

Learn more about the Cortex Query Language `json_path_extract()` function.

##### **IMPORTANT:**

- The `<json_path>` referred to in this JSON function contains new options that aren't available in the other Cortex Query Language (XQL) JSON functions as explained in JSON functions. For more information on the `<json_path>` used in this function, see JSONPath - XPath for JSON.
- Be aware that using this function can be very heavy as it requires a lot of resources to run.

Syntax

Regular Syntax

```
json_path_extract(<json_field>, <json_path>)
```

Syntactic Sugar Format

To make it easier for you to write your XQL queries, you can also use the following syntactic sugar format:

- Using triple quotes:

```
<json_field> ->-> """<json_path>"""
```

- Using single quotes:

```
<json_field> ->-> '<json_path>'
```

Description

The `json_path_extract()` function extracts values from the JSON as defined in the `<json_path>`. This function always returns a string. The syntax used in this function is based on what's used in JavaScript as explained in the jsonpath package.

##### **IMPORTANT:**

JSON field names are case sensitive, so the key to field pairing must be identical in an XQL query for results to be found. For example, if a field value is "TIMESTAMP" and your query is defined to look for "timestamp", no results will be found.

Examples

Defines a JSON object called `Firewall`, which is contained in the JSON file called `json_field`, to extract different string fields called `a`, `b`, `c`, `d`, `e`, `f`, `g`, `h`, and `i`. This example illustrates many different ways you can use the `json_path_extract` function to extract data. Here are how the different fields are configured in the example below to extract data:



- a: Outputs all of the values for the `author` key in the `ServerAccessConfig` JSON array as displayed in column A of the query output below.
- b: Outputs all of the values for the `author` key, where the value for the `priority` key is `22.99` as displayed in column B of the query output below.
- c: Outputs all of the values for the `author` key anywhere found in the JSON as displayed in column C of the query output below.
- d: Outputs all of the values under the `Firewall` key as displayed in column D of the query output below.
- e: Outputs all of the values under the `Firewall` key for the `priority` key as displayed in column E of the query output below.
- f: Outputs the JSON array Index value from the `ServerAccessConfig` JSON array according to its index location from the end of the array as displayed in column F of the query output below.
- g: Outputs all of the JSON array Index values from the `ServerAccessConfig` JSON array according to its index location from the end of the array as displayed in column G of the query output below.
- h: Outputs a specific set of JSON array index values (one or more) from the `ServerAccessConfig` JSON array according to its index location as displayed in column H of the query output below.
- i: Outputs all of the JSON array Index values from the `ServerAccessConfig` JSON array according to its index location from the start (0 Index) to the mentioned index value as displayed in column I of the query output below.

```
dataset = xdr_data | limit 1
| alter
  json_field = "{\"Firewall\": {\"ServerAccessConfig\": [{\"category\": \"policy\", \"author\": \"NRees\", \"name\": \"CustomerSuccess_NoAccess\", \"priority\": 8.95}, {\"category\": \"rule\", \"author\": \"EWaugh\", \"name\": \"AllowAccess_10_10_10\", \"id\": \"0-553-21311-3\", \"priority\": 12.99}, {\"category\": \"rule\", \"author\": \"HMelville\", \"name\": \"SOC_Access\", \"priority\": 8.99}, {\"category\": \"rule\", \"author\": \"JTolkien\", \"name\": \"AllowAccess_JIT\", \"id\": \"0-395-19395-8\", \"priority\": 22.99}], \"Reviewer\": {\"UserName\": \"jdow\", \"Role\": \"Admin\"}}}"
| alter a = json_path_extract(json_field, "$.Firewall.ServerAccessConfig[*].author")
| alter b = json_path_extract(json_field, "$..*[?(@.priority==22.99)].author")
| alter c = json_path_extract(json_field, "$..author")
| alter d = json_path_extract(json_field, "$.Firewall.*")
| alter e = json_path_extract(json_field, "$.Firewall..priority")
| alter f = json_path_extract(json_field, "$..ServerAccessConfig[0.length-1]")
| alter g = json_path_extract(json_field, "$..ServerAccessConfig[-1:]")
| alter h = json_path_extract(json_field, "$..ServerAccessConfig[0,1]")
| alter i = json_path_extract(json_field, "$..ServerAccessConfig[2:]")
| fields json_field, a, b, c, d, e, f, g, h, i
```

Sample output of query

-TIME	JSON_FIELD	A	B	C	D	E	F	G
Jan 20th 2025 18:51:42	{ "Firewall": { "ServerAccessConfig": [ { "category": "policy", "author": "NRees", "name": "CustomerSuccess_NoAccess", "priority": 8.95 }, { "category": "rule", "author": "EWaugh", "name": "AllowAccess_10_10_10", "id": "0-553-21311-3", "priority": 12.99 }, { "category": "rule", "author": "HMelville", "name": "SOC_Access", "priority": 8.99 }, { "category": "rule", "author": "JTolkien", "name": "AllowAccess_JIT", "id": "0-395-19395-8", "priority": 22.99 } ], "Reviewer": { "UserName": "jdow", "Role": "Admin" } }}	NRees, EWaugh, HMelville, JTolkien	JTolkien	NRees, EWaugh, HMelville ,JTolkien	{ "ServerAccessConfig": [ { "category": "policy", "author": "NRees", "name": "CustomerSuccess_NoAccess", "priority": 8.95 }, { "category": "rule", "author": "EWaugh", "name": "AllowAccess_10_10_10", "id": "0-553-21311-3", "priority": 12.99 }, { "category": "rule", "author": "HMelville", "name": "SOC_Access", "priority": 8.99 }, { "category": "rule", "author": "JTolkien", "name": "AllowAccess_JIT", "id": "0-395-19395-8", "priority": 22.99 } ], "Reviewer": { "UserName": "jdow", "Role": "Admin" } }	8.95, 12.99, 8.99, 22.99	[ { "category": "rule", "author": "JTolkien", "name": "AllowAccess_JIT", "id": "0-395-19395-8", "priority": 22.99 }, { "category": "rule", "author": "JTolkien", "name": "AllowAccess_JIT", "id": "0-395-19395-8", "priority": 22.99 } ]	[ { "category": "rule", "author": "JTolkien", "name": "AllowAccess_JIT", "id": "0-395-19395-8", "priority": 22.99 } ]



## 6.4.52 | lag

### Abstract

Learn more about the Cortex Query Language `lag()` navigation function that is used with a `windowcomp` stage.

### Syntax

```
windowcomp lag(<field>) [by <field> [,<field>,...]] sort [asc|desc] <field1> [, [asc|desc] <field2>,...] [as <alias>]
```

### Description

The `lag()` function is a navigation function that is used in combination with a `windowcomp` stage. This function is used to return a single value of a field on a preceding row for each row in the group of rows using a combination of the `by` clause and sort (mandatory).

### Example

Retrieve for each event the timestamp of the previous successful login since the last one.

```
preset = authentication_story
| filter auth_identity not in (null, "") and auth_outcome = "SUCCESS"
| alter ep = to_epoch(_time)
| limit 100
| windowcomp lag(_time) by auth_identity sort asc ep as previous_login
```

## 6.4.53 | last

### Abstract

Learn more about the Cortex Query Language `last` aggregate comp function that returns the last field value found in the dataset with the matching criteria.

### Syntax

```
comp last(<field>) [as <alias>] by <field_1>,<field_2> [addrawdata = true|false [as <target field>]]
```

### Description

The `last` aggregation is a `comp` function that returns the last value found for a field in the dataset over a group of rows that has matching values for the fields identified in the `by` clause. This function is dependent on a time-related field, so for your query to be considered valid, ensure that the dataset running this query contains a time-related field. This function is used in combination with a `comp` stage.

In addition, you can configure whether the raw data events are displayed by setting `addrawdata` to either `true` or `false` (default), which are used to configure the final `comp` results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

### Examples

Return the last timestamp found in the dataset for any given `action_total_download` value for all records that have matching values for their `actor_process_image_path` and `actor_process_command_line` fields. The query calculates a maximum of 100 `xdr_data` records and includes a `raw_data` column listing the raw data events used to display the final `comp` results.

```
dataset = xdr_data
| fields _time, actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp last(_time) as download_time by Process_Path, Process_CMD addrawdata = true as raw_data
```

## 6.4.54 | last\_value

### Abstract

Learn more about the Cortex Query Language `last_value()` navigation function that is used with a `windowcomp` stage.

### Syntax

```
windowcomp last_value(<field>) [by <field> [,<field>,...]] sort [asc|desc] <field1> [, [asc|desc] <field2>,...] [between 0|null|<number>|-<number> [and 0|null|
<number>|-<number>] [frame_type=range]] [as <alias>]
```

### Description

The `last_value()` function is a navigation function that is used in combination with a `windowcomp` stage. This function is used to return a single value of a field for the last row of each row in the group of rows in the current window frame, for all records that contain matching values for the fields identified using a combination of the `by` clause, `sort` (mandatory), and `between` window frame clause.



## Example

Return the last IP address a user authenticated from successfully.

```
preset = authentication_story
| filter auth_identity not in (null, "") and auth_outcome = ""SUCCESS"" and action_country != UNKNOWN
| alter et = to_epoch(_time), t = _time
| bin t span = 1d
| limit 100
| windowcomp last_value(action_local_ip) by auth_identity, t sort asc et between null and null as first_action_local_ip
| fields auth_identity , *action_local_ip
```

## 6.4.55 | latest

### Abstract

Learn more about the Cortex Query Language **latest** aggregate comp function that returns the latest field value found with the matching criteria.

### Syntax

```
comp latest(<field>) [as <alias>] by <field_1>,<field_2> [addrawdata = true|false [as <target field>]]
```

### Description

The **latest** aggregation is a comp function that returns a single chronologically latest value found for a field over a group of rows that has matching values for the fields identified in the **by** clause. This function is dependent on a time-related field, so for your query to be considered valid, ensure that the dataset running this query contains a time-related field. This function is used in combination with a **comp** stage.

In addition, you can configure whether the raw data events are displayed by setting **addrawdata** to either **true** or **false** (default), which are used to configure the final **comp** results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

### Examples

Return the chronologically latest timestamp found for any given **action\_total\_download** value for all records that have matching values for their **actor\_process\_image\_path** and **actor\_process\_command\_line** fields. The query calculates a maximum of 100 **xdr\_data** records and includes a **raw\_data** column listing the raw data events used to display the final **comp** results.

```
dataset = xdr_data
| fields _time, actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp latest(_time) as download_time by Process_Path, Process_CMD addrawdata = true as raw_data
```

## 6.4.56 | len

### Abstract

Learn more about the Cortex Query Language **len** function that returns the number of characters contained in a string.

### Syntax

```
len (<string>)
```

### Description

The **len()** function returns the number of characters contained in a string.

### Examples

Show domain names that are more than 100 characters in length.

```
dataset = xdr_data
| fields dns_query_name
| filter len(dns_query_name) > 100
| limit 10
```

## 6.4.57 | list

### Abstract

Learn more about the Cortex Query Language **list** aggregate comp function that returns an array for up to 100 values for a field in the result set.



## Syntax

```
comp list(<field>) [as <alias>] by <field_1>,<field_2> [addrawdata = true|false [as <target field>]]
```

## Description

The **list** aggregation is a comp function that returns a single array of up to 100 values found for a given field over a group of rows, for all records that contain matching values for the fields identified in the **by** clause. The array values are all non-null, so null values are filtered out. The values returned in the array are non-unique, so if a value repeats multiple times it is included as part of the list of up to 100 values. This function is used in combination with a **comp** stage.

In addition, you can configure whether the raw data events are displayed by setting **addrawdata** to either **true** or **false** (default), which are used to configure the final **comp** results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

## Examples

Return an array containing up to 100 values seen for the **action\_total\_download** field over a group of rows, for all records that have matching values for their **actor\_process\_image\_path** and **actor\_process\_command\_line** values. The query calculates a maximum of 100 **xdr\_data** records and includes a **raw\_data** column listing the raw data events used to display the final **comp** results.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download Download
| filter Download > 0
| limit 100
| comp list(Download) as list_download by Process_Path, Process_CMD addrawdata = true as raw_data
```

## 6.4.58 | **lowercase**

### Abstract

Learn more about the Cortex Query Language **lowercase()** function that converts a string field to all lowercase letters.

## Syntax

```
lowercase (<string>)
```

## Description

The **lowercase()** function converts a string field value to all lowercase.

## Examples

Convert all **actor\_process\_image\_name** field values that are not null to lowercase, and return a list of unique values.

```
dataset = xdr_data
| fields actor_process_image_name as apin
| dedup apin by asc _time
| filter apin != null
| alter apin = lowercase(apin)
```

## 6.4.59 | **ltrim, rtrim, trim**

### Abstract

Learn more about the Cortex Query Language **ltrim()**, **rtrim()**, and **trim()** functions that remove **trim\_characters** from a string.

## Syntax

```
trim (<string>,[trim_characters])
rtrim (<string>,[trim_characters])
ltrim (<string>,[trim_characters])
```

## Description

- The **trim()** function removes all instances of the specified **trim\_characters** defined in the second parameter of the function from the beginning and end of the string defined in the first parameter of the function.
- The **rtrim()** function removes all instances of the specified **trim\_characters** defined in the second parameter of the function from the end of the string defined in the first parameter of the function.
- The **ltrim()** function removes all instances of the specified **trim\_characters** defined in the second parameter of the function from the beginning of the string defined in the first parameter of the function.

Keep in mind the following important points before using these functions, where relevant examples are provided:



- The specified `trim_characters` do not need to be in any order.

Example 116.

Either of these yield the same result:

```
rtrim("explorer.exe", ".ex")
rtrim("explorer.exe", ".exe")
rtrim("explorer.exe", "x.e")
```

Output result:

```
"explorer"
```

- `trim_characters` don't support regular expressions or escape characters.

Example 117.

```
ltrim("***a*aapple*", "*a")
```

Output results:

```
"pple*"
```

- All occurrences of the `trim_characters` supplied are removed from left to right until it reaches a letter that is not part of the supplied letters.

Example 118.

```
ltrim("hello world", "leh")
```

Output results:

```
"o world"
```

- A space in the string can also be considered a character.

- The `ltrim()`, `rtrim()`, and `trim()` functions are case sensitive unless there is an override.

Example 119.

```
ltrim("***a*aapple*", "*A")
```

Output results:

```
"a*aapple*"
```

- If you do not specify `trim_characters`, then whitespace (spaces and tabs) are removed.

Example 120.

```
ltrim(" apple*")
```

Output results:

```
"apple*"
```

## Examples

A complete query example, where the output results of each `ltrim()`, `rtrim()`, and `trim()` function is detailed in the comments.

```
config timeframe = 1w | dataset = xdr_data
| limit 1
| alter
example_1 = rtrim("explorer.exe", "x.e"), // ---> "explorer"
example_2 = ltrim("hello world", "leh"), // ---> "o world"
example_3 = trim(" apple* ", " "), // ---> "apple*"
example_4 = ltrim("***a*aapple*", "*A") // ---> "a*aapple*"
| fields example*
```

## 6.4.60 | max

### Abstract

Learn more about the Cortex Query Language `max` function used with both `comp` and `windowcomp` stages.



## Syntax

comp stage

```
comp max(<field>) [as <alias>] by <field_1>,<field_2> [addraddata = true|false [as <target field>]]
```

windowcomp stage

```
windowcomp max(<field>) [by <field> [,<field>,...]] [sort {asc|desc} <field1> [, {asc|desc} <field2>,...]] [between 0|null|<number>|-<number> [and 0|null|<number>|-<number>] [frame_type=range]] [as <alias>]
```

## Description

The **max()** function is used to return the maximum value of an integer field over a group of rows. The function syntax and application is based on the preceding stage:

comp stage

When the **max** aggregation function is used with a comp stage, the function returns a single maximum value of an integer field for a group of rows, for all records that contain matching values for the fields identified in the **by** clause.

In addition, you can configure whether the raw data events are displayed by setting **addraddata** to either **true** or **false** (default), which are used to configure the final **comp** results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

windowcomp stage

When the **max** aggregate function is used with a windowcomp stage, the function returns a single maximum value of an integer field for each row in the group of rows, for all records that contain matching values for the fields identified using a combination of the **by** clause, **sort**, and **between** window frame clause. The results are provided in a new column in the results table.

## Examples

comp example

Return a single maximum value of the **action\_total\_download** field for a group of rows, for all records that have matching values for their **actor\_process\_image\_path** and **actor\_process\_command\_line** values. The query calculates a maximum of 100 **xdr\_data** records and includes a **raw\_data** column listing a single value for the results.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp max(Download) as max_download by Process_Path, Process_CMD addraddata = true as raw_data
```

windowcomp example

Return the last login time. The query returns a maximum of 100 **authentication\_story** records in a column called **action\_user\_agent**.

```
preset = authentication_story
| limit 100
| windowcomp max(_time) by action_user_agent
```

## 6.4.61 | median

### Abstract

Learn more about the Cortex Query Language **median** function used with both **comp** and **windowcomp** stages.

## Syntax

comp stage

```
comp median(<field>) [as <alias>] by <field_1>,<field_2> [addraddata = true|false [as <target field>]]
```

windowcomp stage

```
windowcomp median(<field>) [by <field> [,<field>,...]] [as <alias>]
```

## Description

The **median()** function is used to return the median value of a field over a group of rows. The function syntax and application is based on the preceding stage:

comp stage

When the **median** aggregation function is used with a comp stage, the function returns a single median value of a field for a group of rows, for all records that contain matching values for the fields identified in the **by** clause.



In addition, you can configure whether the raw data events are displayed by setting `addraddata` to either `true` or `false` (default), which are used to configure the final `comp` results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events. `windowcomp` stage

When the `median` aggregate function is used with a `windowcomp` stage, the function returns a single median value of a field for each row in the group of rows, for all records that contain matching values for the fields identified in the `by` clause. In a `median` function, the `sort` and `between` window frame clause are not used. The results are provided in a new column in the results table.

#### Examples

##### comp example

Return a single median value of the `action_total_download` field over a group of rows, for all records that have matching values for their `actor_process_image_path` and `actor_process_command_line` values. The query calculates a maximum of 100 `xdr_data` records and includes a `raw_data` column listing a single value for the results.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp median(Download) as median_download by Process_Path, Process_CMD
addraddata = true as raw_data
```

##### windowcomp example

Return all events where the `Download` field is greater than the median by reviewing each individual event and how it compares to the median. The query returns a maximum of 100 `xdr_data` records in a column called `median_download`.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| windowcomp median(Download) by Process_Path, Process_CMD as median_download
| filter Download > median_download
```

## 6.4.62 | min

#### Abstract

Learn more about the Cortex Query Language `min` function used with both `comp` and `windowcomp` stages.

#### Syntax

##### comp stage

```
comp min(<field>) [as <alias>] by <field_1>,<field_2> [addraddata = true|false [as <target field>]]
```

##### windowcomp stage

```
windowcomp min(<field>) [by <field> [,<field>,...]] [sort [asc|desc] <field1> [, [asc|desc] <field2>,...]] [between 0|null|<number>|-<number> [and 0|null|
<number>|-<number>] [frame_type=range]] [as <alias>]
```

#### Description

The `min()` function is used to return the minimum value of an integer field over a group of rows. The function syntax and application is based on the preceding stage:

##### comp stage

When the `min` aggregation function is used with a `comp` stage, the function returns a single minimum value of an integer field for a group of rows, for all records that contain matching values for the fields identified in the `by` clause.

In addition, you can configure whether the raw data events are displayed by setting `addraddata` to either `true` or `false` (default), which are used to configure the final `comp` results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

##### windowcomp stage

When the `min` aggregate function is used with a `windowcomp` stage, the function returns a single minimum value of an integer field for each row in the group of rows, for all records that contain matching values for the fields identified using a combination of the `by` clause, `sort`, and `between` window frame clause. The results are provided in a new column in the results table.

#### Examples

##### comp example

Return a single minimum value of the `action_total_download` field for a group of rows, for all records that have matching values for their `actor_process_image_path` and `actor_process_command_line` values. The query calculates a maximum of 100 `xdr_data` records and includes a `raw_data` column listing a single value for the results.



```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp min(Download) as min_download by Process_Path, Process_CMD addrawdata = true as raw_data
windowcomp example
```

Return the first login time. The query returns a maximum of 100 `authentication_story` records in a column called `action_user_agent`.

```
preset = authentication_story
| limit 100
| windowcomp min(_time) by action_user_agent
```

#### 6.4.63 | `multiply`

##### Abstract

Learn more about the Cortex Query Language `multiply()` function that multiplies two integers.

##### Syntax

```
multiply (<string> | <integer>, <string> | <integer>)
```

##### Description

The `multiply()` function multiplies two positive integers. Parameters can be either integer literals, or integers as a string type such as might be contained in a data field.

##### Example

```
dataset = xdr_data
| alter mynum = multiply(action_file_size, 3)
| fields action_file_size, mynum
| filter action_file_size > 0
| limit 1
```

#### 6.4.64 | `object_create`

##### Abstract

Learn more about the Cortex Query Language `object_create()` function.

##### Syntax

```
object_create ("<key1>", "<value1>", "<key2>", "<value2>","...")
```

##### Description

The `object_create()` function returns an object based on the given parameters defined for the key and value pairs. Accepts n > 1 even number of parameters.

##### Example

Returns a final object to a field called `a` that contains the key and value pair {`<key> :<password>`}, where the "password" value is comprised by joining 2 values together.

```
dataset = xdr_data
| alter a = object_create("2", concat("pass", "word"))
| fields a
```

#### 6.4.65 | `object_merge`

##### Abstract

Learn more about the Cortex Query Language `object_merge()` function.

##### Syntax

```
object_merge(<obj1>, <obj2>, <obj3>, ...)
```

##### Description

The `object_merge()` function returns a new object, which is created from a merge of a number of objects. When there is a key name that is duplicated in any of the objects, the value in the new object is determined by the latter argument.



## Example

Two objects are created and merged, where some key names are duplicated, including `name`, `last_name`, and `age`. Since the `name` value is the same for both objects, the same name is used in the new object. Yet, the `last_name` and `age` key values differ, so the values from the second object are used in the new object.

```
dataset = xdr_data
| alter
obj1 = object_create("name", "jane", "last_name", "doe", "age", 33,
obj2 = object_create("name", "jane", "last_name", "simon", "age", 34, "city", "new-york")
| alter result = object_merge(obj1, obj2)
| fields result
```

The function returns the following new object in the RESULT column of the results table:

```
{"name": "jane", "last_name": "simon", "age": 34, "city": "new-york"}
```

## 6.4.66 | parse\_epoch

### Abstract

Learn more about the Cortex Query Language `parse_epoch()` function that returns a Unix epoch TIMESTAMP object.

### Syntax

```
parse_epoch("<format string>", <timestamp field>[, "<time zone>"] [<time unit>])
```

### Description

The `parse_epoch()` function returns a Unix epoch TIMESTAMP object after converting a string representation of a timestamp. The `<time zone>` offset is optional to configure using an hours offset, such as `+08:00`, or using a time zone name from the List of Supported Time Zones, such as `"America/Chicago"`. When you do not configure a timezone, the default is `UTC`. The `<time unit>` is optional to configure and indicates whether the Unix epoch integer value represents seconds, milliseconds, or microseconds. These values are supported, and the default is used when none is configured:

- SECONDS (default)
- MILLIS
- MICROS

### IMPORTANT:

The order of the `<time zone>` and `<time unit>` matters. The `<time zone>` must be defined first followed by the `<time unit>`. If the `<time zone>` is set after the `<time unit>`, the default time zone is used and the configured value is ignored.

### Examples

- With a time zone configured:

Returns a maximum of 100 `xdr_data` records, which includes a timestamp field called `new_time` in the format `MMM dd YYYY HH:mm:ss`, such as `Dec 25th 2008 04:30:00`. This `new_time` field is comprised by taking a character string representation of a timestamp `"Thu Dec 25 07:30:00 2008"` and adding to it `+03:00` hours as the time zone format. This string timestamp is then converted to a Unix epoch TIMESTAMP object in milliseconds using the `parse_epoch` function, and this resulting value is converted to the final timestamp using the `to_timestamp` function.

```
dataset = xdr_data
| alter new_time = to_timestamp(parse_epoch("%c", "Thu Dec 25 07:30:00 2008", "+3", "millis"))
| fields new_time
| limit 100
```

- Without a time zone or time unit configured:

Returns a maximum of 100 `xdr_data` records, which includes a timestamp field called `new_time` in the format `MMM dd YYYY HH:mm:ss`, such as `Dec 25th 2008 04:30:00`. This `new_time` field is comprised by taking a character string representation of a timestamp `"Thu Dec 25 07:30:00 2008"` and adding to it a UTC time zone format (default when none configured). This string timestamp is then converted to a Unix epoch TIMESTAMP object in seconds (default when none configured) using the `parse_epoch` function, and this resulting value is converted to the final timestamp using the `to_timestamp` function.

```
dataset = xdr_data
| alter new_time = to_timestamp(parse_epoch("%c", "Thu Dec 25 07:30:00 2008"))
| fields new_time
| limit 100
```

## 6.4.67 | parse\_timestamp

### Abstract



Learn more about the Cortex Query Language `parse_timestamp()` function that returns a `TIMESTAMP` object.

#### Syntax

```
parse_timestamp("<format time string>", "<time string>" | format_string(<time field>) | <time string field>)
parse_timestamp("<format time string>", "<time string>" | format_string(<time field>) | <time string field>, "<time zone>")
```

#### Description

The `parse_timestamp()` function returns a `TIMESTAMP` object after converting a string representation of a timestamp. The `<time zone>` offset is optional to configure using an hours offset, such as `+08:00`, or using a time zone name from the List of Supported Time Zones, such as "America/Chicago". The `parse_timestamp()` function can include both an alter stage and `format_string` function. For more information, see the examples below. The `format_string` function contains the format elements that define how the `parse_timestamp` string is formatted. Each element in the `parse_timestamp` string must have a corresponding element in `format_string`. The location of each element in the `format_string` must match the location of each element in `parse_timestamp`.

#### Examples

- Without a time zone configured

Returns a maximum of 100 `microsoft_dhcp_raw` records, which includes a `TIMESTAMP` object in the `p_t_test` field in the format `MMM dd YYYY HH:mm:ss`, such as Jun 25th 2021 18:31:25. This format is detailed in the `format_string` function, which includes merging both the `date` and `time` fields.

```
dataset = microsoft_dhcp_raw
| alter p_t_test = parse_timestamp("%m/%d/%Y %H:%M:%S", format_string("%s %s", date, time))
| fields p_t_test
| limit 100
```

- With a time zone name configured

Returns a maximum of 100 `microsoft_dhcp_raw` records, which includes a `TIMESTAMP` object in the `p_t_test` field in the format `MMM dd YYYY HH:mm:ss`, such as Jun 25th 2021 18:31:25. This format is detailed in the `format_string` function, which includes merging both the `date` and `time` fields, and includes a "Asia/Singapore" time zone.

```
dataset = microsoft_dhcp_raw
| alter p_t_test = parse_timestamp("%m/%d/%Y %H:%M:%S", format_string("%s %s", date, time), "Asia/Singapore")
| fields p_t_test
| limit 100
```

- With a time zone configured using an hours offset

Returns a maximum of 100 `microsoft_dhcp_raw` records, which includes a `TIMESTAMP` object in the `p_t_test` field in the format `MMM dd YYYY HH:mm:ss`, such as Jun 25th 2021 18:31:25. This format is detailed in the `format_string` function, which includes merging both the `date` and `time` fields, and includes a time zone using an hours offset of "+08:00".

```
dataset = microsoft_dhcp_raw
| alter p_t_test = parse_timestamp("%m/%d/%Y %H:%M:%S", format_string("%s %s", date, time), "+08:00")
| fields p_t_test
| limit 100
```

- Convert a time string that contains milliseconds

Returns a single `xdr_data` record, which includes both, a manually added time string, "Jun 25 2024 18:31:25.723", in the `time_string` field and a `TIMESTAMP` object in the `p_t_test` field, such as Jun 25 2024 18:31:25, as the result of the `parse_timestamp()` function. Notice that the format element `%E*S` is used to capture seconds including any level of fractional precision, such as milliseconds.

```
dataset = xdr_data
| limit 1
| alter time_string = "Jun 25 2024 18:31:25.723"
| alter p_t_test = parse_timestamp("%h %d %Y %H:%M:%E3S", time_string)
| fields p_t_test, time_string
```

## 6.4.68 | `pow`

#### Abstract

Learn more about the Cortex Query Language `pow()` function that returns the value of a number raised to the power of another number.

#### Syntax

```
pow (<x,n>)
```

#### Description

The `pow()` function returns the value of a number (`x`) raised to the power of another number (`n`).



## 6.4.69 | rank

### Abstract

Learn more about the Cortex Query Language `rank()` numbering function that is used with a `windowcomp` stage.

### Syntax

```
windowcomp rank() [by <field> [,<field>,...]] sort [asc|desc] <field1> [, [asc|desc] <field2>,...] [as <alias>]
```

### Description

The `rank()` function is a numbering function that is used in combination with a `windowcomp` stage. This function is used to return a single value for the ordinal (1-based) rank for each row in the group of rows using a combination of the `by` clause and `sort` (mandatory).

### Example

Return an average ranking for the average CPU usage on `metric_type=HOST`. Allows you to see changes in the CPU usage compared to all hosts in the environment. The query returns a maximum of 100 `it_metrics` records. The results are ordered by `ft` in descending order in the `rank` column.

```
dataset = it_metrics
| filter metric_type = HOST
| alter cpu_avg_str = to_string(cpu_avg)
| alter ft = date_floor(_time, "w")
| alter dt = date_floor(_time, "d")
| limit 100
| windowcomp rank() by ft sort desc cpu_avg_str as rank
| filter (agent_hostname contains $host_name)
| comp avg(rank) by dt
```

## 6.4.70 | regexcapture

### Abstract

Learn more about the Cortex Query Language `regexcapture()` function used in Parsing Rules to extract data from fields using regular expression named groups from a given string.

### IMPORTANT:

The `regexcapture()` function is only supported in the XQL syntax for Parsing Rules.

### Syntax

```
regexcapture(<field>, "<pattern>")
```

### Description

In Parsing Rules, the `regexcapture()` function is used to extract data from fields using regular expression named groups from a given string and returns a JSON object with captured groups. This function can be used in any section of a Parsing Rule. The `regexcapture()` function is useful when the regex pattern is not identical throughout the log, which is required when using the `reextract` function.

XQL uses RE2 for its regular expression implementation. When using the `(?1)` syntax for case-insensitive mode in your query, this syntax should be added only once at the beginning of the inline regular expression.

### Example

Parsing Rule to create a dataset called `my_regexcapture_test`, where the vendor and product that the specified Parsing Rules applies to is called `regexcapture_vendor` and `regexcapture_product`. The output results includes a new field called `regexcaptureResult`, which extract data from the `_raw_log` field using regular expression named groups as defined and returns the captured groups.

Parsing Rule:

```
[INGEST:vendor="regexcapture_vendor", product="regexcapture_product", target_dataset="my_regexcapture_test"]
alter regexcaptureResult = regexcapture(_raw_log,"^(?P<ip>\d{1,3}.\d{1,3}.\d{1,3}.\d{1,3}) - (?P<user>\w+) \[(?P<timestamp>.+)\] (?P<request>.+) (?P<status>\d{3}) (?P<bytes>\d+)" );
```

Log:

```
192.168.1.1 - john [10/Mar/2024:12:34:56 +0000] GET /index.html HTTP/1.1 200 1234
```

XQL Query:

For the `my_regexcapture_test` dataset, returns the `regexcaptureResult` field output.

```
dataset = my_regexcapture_test
| fields regexcaptureResult
```

regexcaptureResult field output:



```
{
  "ip": "192.168.1.1",
  "user": "john",
  "timestamp": "10/Mar/2024:12:34:56 +0000",
  "request": "GET /index.html HTTP/1.1",
  "status": "200",
  "bytes": "1234"
}
```

#### 6.4.71 | `regextract`

##### Abstract

Learn more about the Cortex Query Language `regextract()` function that uses regular expressions to assemble an array of matching substrings from a string.

##### Syntax

```
regextract (<string_value>, <pattern>)
```

##### Description

The `regextract()` function accepts a string and a regular expression, and it returns an array containing substrings that match the expression.

Cortex Query Language (XQL) uses RE2 for its regular expression implementation. While capturing multiple groups is unsupported, capturing one group in queries is supported.

When using the `(?i)` syntax for case-insensitive mode in your query, this syntax should be added only once at the beginning of the inline regular expression.

##### **NOTE:**

Capturing multiple groups is supported in Parsing Rules when using the `regexcapture` function.

##### Examples

###### Without a capturing group

Extract the `Account Name` from the `action_evtlog_message`. Use the `arrayindex` and `split` functions to extract the actual account name from the array created by `regextract`.

```
dataset = xdr_data
| fields action_evtlog_message as aem
| filter aem != null
| alter account_name =
  arrayindex(
    split(
      arrayindex(
        regextract(aem, "Account Name:\t\t.*\r\n")
        , 0)
      , ":")
    , 1)
| filter account_name != null
| limit 10
```

###### Using one capturing group

Extract from the `log_example` field all of the values included for the `id` objects.

```
dataset = xdr_data
| limit 1
| alter
  log_example = "{\"events\": [{\"id\": \"1\", \"type\": \"process\", \"size\": 123, \"processID\": 40540}, {\"id\": \"2\", \"type\": \"request\", \"size\": 456,
\"srcOS\": \"MAC\"}, {\"host\": \"LocalHost\", \"date\": {\"day\": 4, \"month\": 7, \"year\": 2024}, \"tags\": [\"agent\", \"auth\", \"low\"]}]"
| alter
  one_capture_group_usage = regextract(log_example, "\"id\":\\s*(\\{[^\\"]+\\})\\\"")
| fields log_example, one_capture_group_usage
```

#### 6.4.72 | `replace`

##### Abstract

Learn more about the Cortex Query Language `replace()` function that performs a substring replacement.

##### Syntax

```
replace (<field>, "<old_substring>", "<new_string>")
```

##### Description

The `replace()` function accepts a string field, and replaces all occurrences of a substring with a replacement string.



## Examples

If '.exe' is present on the **action\_process\_image\_name** field value, replace that substring with an empty string. This example uses the if and lowercase functions, as well as the contains operator to perform the conditional check.

```
dataset = xdr_data
| fields action_process_image_name as apin
| filter apin != null
| alter remove_exe_process = if(lowercase(apin) contains ".exe",
    replace(lowercase(apin),".exe","");
    lowercase(apin))
| limit 10
```

See also the ltrim, rtrim, trim function example.

## 6.4.73 | **replex**

### Abstract

Learn more about the Cortex Query Language **replex()** function that uses a regular expression to identify and replace substrings.

### Syntax

```
replex (<string>, <pattern>, <new_string>)
```

### Description

The **replex()** function accepts a string, and then uses a regular expression to identify a substring, and then replaces matching substrings with a new string.

XQL uses RE2 for its regular expression implementation.

### Examples

For any **agent\_id** that contains a dotted decimal IP address, mask the IP address. Use the dedup stage to reduce the result set to first-seen **agent\_id** values.

```
dataset = xdr_data
| fields agent_id
| alter clean_agent_id = replex(agent_id,
    "[\d]+\.[\d]+\.[\d]+\.[\d]+",
    "xxx.xxx.xx.xx")
| dedup agent_id by asc _time
```

## 6.4.74 | **round**

### Abstract

Learn more about the Cortex Query Language **round()** function that returns the input value rounded to the nearest integer.

### Syntax

```
round (<float> | <integer>)
```

### Description

The **round()** function accepts either a float or an integer as an input value, and it returns the input value rounded to the nearest integer.

### Example

```
dataset = xdr_data
| alter mynum = divide(action_file_size, 7)
| alter mynum2 = round(mynum)
| fields action_file_size, mynum, mynum2
| filter action_file_size > 3
| limit 1
```

## 6.4.75 | **row\_number**

### Abstract

Learn more about the Cortex Query Language **row\_number()** numbering function that is used with a **windowcomp** stage.

### Syntax

```
windowcomp row_number() [by <field> [<field>, ...]] [sort [asc|desc] <field1> [, [asc|desc] <field2>, ...]] [as <alias>]
```



## Description

The **row\_number()** function is a numbering function that is used in combination with a windowcomp stage. This function is used to return a single value for the sequential row ordinal (1-based) for each row from a group of rows using a combination of the **by** clause and **sort**.

## Example

Return a single value for the sequential row ordinal (1-based) for each row in the group of rows. The query returns a maximum of 100 **xdr\_data** records. The results are ordered by the **source\_ip** in ascending order in the **row\_number\_dns\_query\_name** column.

```
dataset = xdr_data
| limit 100
| windowcomp row_number() sort source_ip as row_number_dns_query_name
```

## 6.4.76 | **split**

### Abstract

Learn more about the Cortex Query Language **split()** function that splits a string and returns an array of string parts.

### Syntax

```
split (<value> [, <string_delimiter>])
```

### Description

The **split()** function splits a string using an optional delimiter, and returns the resulting substrings in an array. If no delimiter is specified or an empty string is specified as a delimiter, a space (' ') is used.

### Examples

Split IP addresses into an array, each element of the array containing an IP octet.

```
dataset = xdr_data
| fields action_local_ip as alii
| alter ip_octets = split(alii, ".")
| limit 10
```

## 6.4.77 | **stddev\_population**

### Abstract

Learn more about the Cortex Query Language **stddev\_population()** function used with both **comp** and **windowcomp** stages.

### Syntax

#### comp stage

```
comp stddev_population(<field>) [as <alias>] by <field_1>,<field_2> [addraddata = true|false [as <target field>]]
```

#### windowcomp stage

```
windowcomp stddev_population(<field>) [by <field> [,<field>,...]] [sort [asc|desc] <field1> [, [asc|desc] <field2>,...]] [between 0|null|<number>|-<number> [and 0|null|<number>|-<number>] [frame_type=range]] [as <alias>]
```

### Description

The **stddev\_population()** function is used to return a single population (biased) variance value of a field for a group of rows. The function syntax and application is based on the preceding stage:

#### comp stage

When the **stddev\_population** aggregation function is used with a comp stage, the function returns a single population (biased) variance value of a field over a group of rows, for all records that contain matching values for the fields identified in the **by** clause.

In addition, you can configure whether the raw data events are displayed by setting **addraddata** to either **true** or **false** (default), which are used to configure the final **comp** results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

#### windowcomp stage

When the **stddev\_population** statistical aggregate function is used with a windowcomp stage, the function returns a single population (biased) variance value of a field for each row in the group of rows, for all records that contain matching values for the fields identified using a combination of the **by** clause, **sort**, and **between** window frame clause. The results are provided in a new column in the results table.



## Examples

### comp example

Calculates a maximum of 100 `metrics_source` records, where the `_broker_device_id` is `655AYUWF`, and include a single population (biased) variance value of the `total_size_rate` field for a group of rows.

```
dataset = metrics_source
| filter _broker_device_id = "655AYUWF"
| comp stddev_population(total_size_rate)
| limit 100
```

### windowcomp example

Return maximum of 100 `metrics_source` records and include a single population (biased) variance value of the `total_size_rate` field for each row in the group of rows, for all records that contain matching values in the `_broker_device_id` field. The results are provided in the `stddev_population` column.

```
dataset = metrics_source
| limit 100
| windowcomp stddev_population(total_size_rate) by _broker_device_id as `stddev_population`
```

## 6.4.78 | `stddev_sample`

### Abstract

Learn more about the Cortex Query Language `stddev_sample()` function used with both `comp` and `windowcomp` stages.

### Syntax

#### comp stage

```
comp stddev_sample(<field>) [as <alias>] by <field_1>,<field_2> [addradata = true|false [as <target field>]]
```

#### windowcomp stage

```
windowcomp stddev_sample(<field>) [by <field> [,<field>,...]] [sort [asc|desc] <field1> [, [asc|desc] <field2>,...]] [between 0|null|<number>|-<number> [and 0|null|<number>|-<number>] [frame_type=range]] [as <alias>]
```

### Description

The `stddev_sample()` function is used to return a single sample (unbiased) standard deviation value of a field for a group of rows. The function syntax and application is based on the preceding stage:

#### comp stage

When the `stddev_sample` aggregation function is used with a comp stage, the function returns a single sample (unbiased) standard deviation value of a field over a group of rows, for all records that contain matching values for the fields identified in the `by` clause.

In addition, you can configure whether the raw data events are displayed by setting `addradata` to either `true` or `false` (default), which are used to configure the final `comp` results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

#### windowcomp stage

When the `stddev_sample` statistical aggregate function is used with a windowcomp stage, the function returns a single sample (unbiased) standard deviation value of a field for each row in the group of rows, for all records that contain matching values for the fields identified using a combination of the `by` clause, `sort`, and `between` window frame clause. The results are provided in a new column in the results table.

## Examples

### comp stage example

Calculate a maximum of 100 `metrics_source` records, where the `_broker_device_ip` is `172.16.1.25`, and include a single sample (unbiased) standard deviation value of the `total_size_bytes` field for a group of rows.

```
dataset = metrics_source
| filter _broker_device_ip = "172.16.1.25"
| comp stddev_sample(total_size_bytes)
| limit 100
```

### windowcomp stage example

Return a maximum of 100 `metrics_source` records and include a single sample (unbiased) standard deviation value of the `total_size_rate` field for each row in the group of rows, for all records that contain matching values in the `_broker_device_id` field. The results are provided in the `stddev_sample` column.

```
dataset = metrics_source
| limit 100
| windowcomp stddev_sample(total_size_rate) by _broker_device_id as `stddev_sample`
```



## 6.4.79 | string\_count

### Abstract

Learn more about the Cortex Query Language `string_count()` function that returns the number of times a substring appears in a string.

### Syntax

```
string_count (<string>, <pattern>)
```

### Description

The `string_count()` function returns the number of times a substring appears in a string.

### Example

```
dataset = xdr_data
| fields actor_primary_username as apu
| filter string_count(apu, "e") > 1
```

## 6.4.80 | subtract

### Abstract

Learn more about the Cortex Query Language `subtract()` function that subtracts two integers.

### Syntax

```
subtract (<string1> | <integer1>, <string2> | <integer2>)
```

### Description

The `subtract()` function subtracts two positive integers by subtracting the second argument from the first argument. Parameters may be either integer literals, or integers as a string type such as might be contained in a data field.

### Example

```
dataset = xdr_data
| alter mynum = subtract(action_file_size, 3)
| fields action_file_size, mynum
| filter action_file_size > 3
| limit 1
```

## 6.4.81 | sum

### Abstract

Cortex Query Language `sum` function used with both `comp` and `windowcomp` stages.

### Syntax

#### comp stage

```
comp sum(<field>) [as <alias>] by <field_1>,<field_2> [addraddata = true|false [as <target field>]]
```

#### windowcomp

```
windowcomp sum(<field>) [by <field> [,<field>,...]] [sort [asc|desc] <field1> [, [asc|desc] <field2>,...]] [between 0|null|<number>|-<number> [and 0|null|<number>|-<number>] [frame_type=range]] [as <alias>]
```

### Description

The `sum()` function is used to return the sum of an integer field over a group of rows. The function syntax and application is based on the preceding stage:

#### comp stage

When the `sum` aggregation function is used with a comp stage, the function returns a single sum of an integer field for a group of rows, for all records that contain matching values for the fields identified in the `by` clause.

In addition, you can configure whether the raw data events are displayed by setting `addraddata` to either `true` or `false` (default), which are used to configure the final `comp` results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

#### windowcomp stage

When the `sum` aggregate function is used with a windowcomp stage, the function returns a single sum of an integer field for each row in the group of rows, for all records that contain matching values for the fields identified using a combination of the `by` clause, `sort`, and `between` window frame clause. The results



are provided in a new column in the results table.

#### Examples

comp example

Return a single sum of the `action_total_download` field for a group of rows, for all records that have matching values for their `actor_process_image_path` and `actor_process_command_line` values. The query calculates a maximum of 100 `xdr_data` records and includes a `raw_data` column listing a single value for the results.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp sum(Download) as total_download by Process_Path, Process_CMD addrawdata = true as raw_data
```

windowcomp

Return the download to upload ratio per process. The query returns a maximum of 100 `xdr_data` records in new columns called `sum_upload` and `sum_download`.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download, action_total_upload as Upload
| filter Download > 0
| limit 100
| windowcomp sum(Download) by Process_Path, Process_CMD as sum_download
| windowcomp sum(Upload) by Process_Path, Process_CMD as sum_upload
| fields - Download ,Upload
| dedup Process_CMD, Process_Path, sum_download ,sum_upload
| alter ration = divide(sum_download ,sum_upload)
```

## 6.4.82 | time\_frame\_end

#### Abstract

Learn more about the Cortex Query Language `time_frame_end()` function that returns the end time of the time range specified for the query.

#### Syntax

```
time_frame_end(<time frame>)
```

#### Description

The `time_frame_end()` function returns the timestamp object for the string representation of the end of the time frame configured for the query in the format `MMM dd YYYY HH:mm:ss`, such as Jun 8th 2022 15:20:06. You can configure the time frame using the config `timeframe` function, where the range can be relative or exact.

If the time frame is relative, for example `last 24H`, the function returns the `current_time`. This function is useful when the query uses a custom time frame whose end time is in the past.

#### Example 1 - Relative Time

For the last 5 days from when the query is sent, returns a maximum of 100 `xdr_data` records with the events of the `_time` field with a new field called "x". The "x" field lists the final timestamp at the end of 5 days from when the query was sent for the events in descending order. For more information on this relative timeframe range, see the config `timeframe` function.

```
config timeframe = 5d
| dataset = xdr_data
| alter x = time_frame_end()
| fields x
| sort desc x
```

#### Example 2 - Relative Time

For the last 5 days from when the query is run until now, returns a maximum of 100 `xdr_data` records with the events of the `_time` field with a new field called "x". The "x" field lists the final timestamp at the end of 5 days from when the query runs for the events in descending order. For more information on this relative time frame range, see the config `timeframe` function.

```
config timeframe = between "5d" and "now"
| dataset = xdr_data
| alter x = time_frame_end()
| fields x
| sort desc x
```

## 6.4.83 | timestamp\_diff

#### Abstract



Learn more about the Cortex Query Language `timestamp_diff()` function that returns the difference between two timestamp objects.

#### Syntax

```
timestamp_diff (<timestamp1>, <timestamp2>, <part>)
```

#### Description

The `timestamp_diff()` function returns the difference between two timestamp objects. The units used to express the difference is identified by the `part` parameter. The second timestamp is subtracted from the first timestamp. If the first timestamp is greater than the second, a positive value is returned. If the result of this function is between 0 and 1, 0 is returned.

Supported parts are:

- DAY
- HOUR
- MINUTE
- SECOND
- MILLISECOND
- MICROSECOND

#### Example

```
dataset = xdr_data
| filter story_publish_timestamp != null
| alter ts = to_timestamp(story_publish_timestamp, "MILLIS")
| alter ct = current_time()
| alter diff = timestamp_diff(ct, ts, "MINUTE")
| fields ts, ct, diff
| limit 1
```

### 6.4.84 | `timestamp_seconds`

#### Abstract

Learn more about the Cortex Query Language `timestamp_seconds()` function.

#### Syntax

```
timestamp_seconds (<integer>)
```

#### Description

The `timestamp_seconds()` function converts an epoch time Integer value in seconds to a TIMESTAMP compatible value.

#### NOTE:

Endpoint Detection and Response (EDR) columns store epoch milliseconds values so this function is more useful for values that you insert.

#### Example

Display a human-readable timestamp for the `action_file_access_time` field.

```
alter access_timestamp = timestamp_seconds(1611882205) | limit 1
```

### 6.4.85 | `to_boolean`

#### Abstract

Learn more about the Cortex Query Language `to_boolean()` function that converts a string to a boolean.

#### Syntax

```
to_boolean(<string>)
```

#### Description

The `to_boolean()` function converts a string that represents a boolean to a boolean value.

The input value to this string must be either `TRUE` or `FALSE`, case insensitive.



## 6.4.86 | `to_epoch`

### Abstract

Learn more about the Cortex Query Language `to_epoch()` function that converts a timestamp value for a field or function to the Unix epoch timestamp format.

### Syntax

```
to_epoch (<timestamp>, <time unit>)
```

### Description

The `to_epoch()` function converts a timestamp value for a particular field or function to the Unix epoch timestamp format. This function requires a `<time unit>` value, which indicates whether the integer value for the Unix epoch timestamp format represents seconds (default), milliseconds, or microseconds. If no `<time unit>` is configured, the default is used. Supported values are:

- SECONDS
- MILLIS
- MICROS

### Example

Returns a maximum of 100 `xdr_data` records with the events of the `_time` field, which includes a timestamp field in the Unix epoch format called `ts`. The `ts` field contains the equivalent Unix epoch values in milliseconds for the timestamps listed in the `_time` field.

```
dataset = xdr_data
| filter _time != null
| alter ts = to_epoch(_time, "MILLIS")
| fields ts
| limit 100
```

## 6.4.87 | `to_float`

### Abstract

Learn more about the Cortex Query Language `to_float()` function that converts a string to a floating point number.

### Syntax

```
to_float(<string>)
```

### Description

The `to_float()` function converts a string that represents a number to a floating point number. This function is identical to the `to_number` function.

### Examples

Display the first 10 IP addresses that begin with a value greater than 192. Use the `split` function to split the IP address by '.', and then use the `arrayindex` function to retrieve the first value in the resulting array. Convert this to a number and perform an arithmetic compare to arrive at a result set.

```
dataset = xdr_data
| fields action_local_ip as alii
| filter to_float(arrayindex(split(alii, "."),0)) > 192
| limit 10
```

## 6.4.88 | `to_integer`

### Abstract

Learn more about the Cortex Query Language `to_integer()` function that converts a string field to an integer.

### Syntax

```
to_integer(<string>)
```

### Description

The `to_integer()` function converts a string value that represents a number of a given field to an integer. A good application of using the `to_integer` function is when querying for USB vendor IDs and USB product IDs, which are usually provided in a hex format.

It is an error to provide a string to this function that contains a floating point number.



## Examples

Display the first 10 IP addresses that begin with a value greater than 192. Use the split function to split the IP address by '.', and then use the arrayindex function to retrieve the first value in the resulting array. Convert this to a number and perform an arithmetic compare to arrive at a result set.

```
dataset = xdr_data
| fields action_local_ip as alii
| filter to_integer(arrayindex(split(alii, ".") ,0)) > 192
| limit 10
```

## 6.4.89 | `to_json_string`

### Abstract

Learn more about the Cortex Query Language `to_json_string()` function that accepts all data types and returns its contents as a JSON formatted string.

### Syntax

```
to_json_string(<data type>)
```

### Description

The `to_json_string()` function accepts all data types, such as integers, booleans, strings, and returns it as a JSON formatted string. This function always returns a string. When the input is an object or an array, the function returns a JSON formatted string of the input. When the input string is a string, it returns the string as is. You can then use the JSON formatted string or string returned by this function with the `json_extract`, `json_extract_array`, and `json_extract_scalar` functions.

### Examples

Return the `action_file_device_info` field in JSON format.

```
dataset = xdr_data
| fields action_file_device_info as afdi
| filter afdi != null
| alter the_json_string = to_json_string(afdi)
| limit 10
```

## 6.4.90 | `to_number`

### Abstract

Learn more about the Cortex Query Language `to_number()` function that converts a string to a number.

### Syntax

```
to_number (<string>)
```

### Description

The `to_number()` function converts a string that represents a number to a floating point number. This function is identical to the `to_float` function.

### Examples

Display the first 10 IP addresses that begin with a value greater than 192. Use the split function to split the IP address by '.', and then use the arrayindex function to retrieve the first value in the resulting array. Convert this to a number and perform an arithmetic compare to arrive at a result set.

```
dataset = xdr_data
| fields action_local_ip as alii
| filter to_number(arrayindex(split(alii, ".") ,0)) > 192
| limit 10
```

## 6.4.91 | `to_string`

### Abstract

Learn more about the Cortex Query Language `to_string` function that converts a number value to a string.

### Syntax

```
to_string (<field>)
```



## Description

The `to_string()` function converts a number value of a given field to a string.

## Examples

Display the first non-NULL `action_boot_time` field value. In a second column called `abt_string`, use the concat function to prepend "str: " to the value, and then display it.

```
dataset = xdr_data
| fields action_boot_time as abt
| filter abt != null
| alter abt_string = concat("str: ", to_string(abt))
| limit 1
```

## 6.4.92 | `to_timestamp`

### Abstract

Learn more about the Cortex Query Language `to_timestamp()` function that converts an integer to a timestamp.

### Syntax

```
to_timestamp (<integer>, <units>)
```

### Description

The `to_timestamp()` function converts an integer to a timestamp. This function requires a `units` value, which indicates whether the integer represents seconds, milliseconds, or microseconds since the Unix epoch. Supported values are:

- SECONDS
- MILLIS
- MICROS

### Example

```
dataset = xdr_data
| filter story_publish_timestamp != null
| alter ts = to_timestamp(story_publish_timestamp, "MILLIS")
| fields ts
```

## 6.4.93 | `uppercase`

### Abstract

Learn more about the Cortex Query Language `uppercase()` function that converts a string field to all uppercase letters.

### Syntax

```
uppercase (<string>)
```

### Description

The `uppercase()` function converts a string field value to all uppercase.

## Examples

Convert all `actor_process_image_name` field values that are not null to uppercase, and return a list of unique values.

```
dataset = xdr_data
| fields actor_process_image_name as apin
| dedup apin by asc _time
| filter apin != null
| alter apin = uppercase(apin)
```

## 6.4.94 | `values`

### Abstract

Cortex Query Language `comp values` aggregate returns an array for all the values seen for the field in the result set.



## Syntax

```
comp values(<field>) [as <alias>] by <field_1>,<field_2> [addraddata = true|false [as <target field>]]
```

## Description

The **values** aggregation is a comp function that returns an array of all the values found for a given field over a group of rows, for all records that contain matching values for the fields identified in the **by** clause. The array values are all non-null. Each value appears in the array only once, even if a given value repeats multiple times in the result set. This function is used in combination with a **comp** stage.

In addition, you can configure whether the raw data events are displayed by setting **addraddata** to either **true** or **false** (default), which are used to configure the final **comp** results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

## Example

Return an array containing all the values seen for the **action\_total\_download** field for all records that have matching values for their **actor\_process\_image\_path** and **actor\_process\_command\_line** values. The query calculates a maximum of 100 **xdr\_data** records and includes a **raw\_data** column listing the raw data events used to display the final **comp** results. In addition, this example contains a number of fields defined as aliases: **actor\_process\_image\_path** uses the alias **Process\_Path**, **actor\_process\_command\_line** uses the alias **Process\_CMD**, **action\_total\_download** uses the alias **Download**, and **Download** uses the alias **values\_download**.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp values(Download) as values_download by Process_Path, Process_CMD addraddata = true as raw_data
```

## 6.4.95 | var

### Abstract

Learn more about the Cortex Query Language **var** aggregate comp function that returns the variance value of a field in the result set.

## Syntax

```
comp var(<field>) [as <alias>] by <field_1>,<field_2> [addraddata = true|false [as <target field>]]
```

## Description

The **var** aggregation is a comp function that returns a single variance value of a field over a group of rows, for all records that contain matching values for the fields identified in the **by** clause. This function is used in combination with a **comp** stage.

In addition, you can configure whether the raw data events are displayed by setting **addraddata** to either **true** or **false** (default), which are used to configure the final **comp** results. When including raw data events in your query, the query runs for up to 50 fields that you define and displays up to 100 events.

## Example

Return the variance of the **action\_total\_download** field for all records that have matching values for their **actor\_process\_image\_path** and **actor\_process\_command\_line** values. The query calculates a maximum of 100 **xdr\_data** records and includes a **raw\_data** column listing the raw data events used to display the final **comp** results.

```
dataset = xdr_data
| fields actor_process_image_path as Process_Path, actor_process_command_line as Process_CMD, action_total_download as Download
| filter Download > 0
| limit 100
| comp var(Download) as variance_download by Process_Path, Process_CMD
addraddata = true as raw_data
```

## 6.4.96 | wildcard\_match

### Abstract

Learn more about the Cortex Query Language **wildcard\_match()** function.

## Syntax

```
wildcard_match (<string_value>, <wildcard_pattern>)
```

## Description

The **wildcard\_match()** function accepts a string and a wildcard pattern including \* and ?, and returns **true** when the **<string\_value>** matches the **<wildcard\_pattern>**. When using a wildcard pattern, keep in mind the difference between the following metacharacters used as repetition operators:



- \*: Returns all results following the pattern for a sequence of zero or more (possibly different) strings.
- ?: Returns a single character result following the pattern for a sequence of zero or one in the string.

XQL uses RE2 for its regular expression implementation. When using the (?i) syntax for case-insensitive mode in your query, this syntax should be added only once at the beginning of the inline regular expression.

#### Examples

Lookup dataset sample data

The examples provided below are based on running queries using a lookup dataset called `lookup_app_category`. Here's some sample data of the dataset table:

_TIME	APP_CATEGORY
Aug 31 2024 12:13:48	general-internet
Sep 1st 2024 12:13:11	general-internet
Sep 2th 2024 01:13:17	general-internet
Sep 3th 2024 09:13:37	general-ip
Sep 4th 2024 11:13:25	general-ip
Sep 5th 2024 03:13:19	general-dns
Sep 6th 2024 12:05:03	general-internet
Sep 6th 2024 15:22:36	multicast-ip

#### Example 1: wildcard\_match with filter and \*

Return a maximum of 100 records from a lookup dataset called `lookup_app_category`. The `pattern` field defines the wildcard pattern as `general-*`. When this wildcard pattern is found in the `app_category` field, the `wildcard_match` returns `true` and the matching events are listed in the table results with the `pattern` field displaying the defined pattern.

```
dataset = lookup_app_category
| alter pattern = "general-*"
| filter wildcard_match(app_category , pattern)
| fields pattern, app_category
| limit 100
```

Sample output table results:

PATTERN	_TIME	APP_CATEGORY
general-*	Aug 31 2024 12:13:48	general-internet
general-*	Sep 1st 2024 12:13:11	general-internet
general-*	Sep 2th 2024 01:13:17	general-internet
general-*	Sep 3th 2024 09:13:37	general-ip



PATTERN	_TIME	APP_CATEGORY
general-*	Sep 4th 2024 11:13:25	general-ip
general-*	Sep 5th 2024 03:13:19	general-dns
general-*	Sep 6th 2024 12:05:03	general-internet

Example 2: wildcard\_match with filter and ?

Return a maximum of 100 records from a lookup dataset called `lookup_app_category`. The `pattern` field defines the wildcard pattern as `general-??`. When this wildcard pattern is found in the `app_category` field, the `wildcard_match` returns `true` and the matching events are listed in the table results with the `pattern` field displaying the defined pattern.

```
dataset = lookup_app_category
| alter pattern = "general-??"
| filter wildcard_match(app_category , pattern)
| fields pattern, app_category
| limit 100
```

Sample output table results:

PATTERN	_TIME	APP_CATEGORY
general-??	Sep 3th 2024 09:13:37	general-ip
general-??	Sep 4th 2024 11:13:25	general-ip

Example 3: wildcard\_match with alter, \*, and ?

Return a maximum of 100 records from a lookup dataset called `lookup_app_category`. The `pattern` field defines the wildcard pattern as `*-i??`. When this wildcard pattern is found in the `app_category` field, the `wildcard_match` returns `true` and is passed to the `pattern` field, and the events found are listed in the table results.

```
dataset = lookup_app_category
| alter pattern = "*-i??""
| alter test = wildcard_match(app_category , pattern)
| fields test, pattern, app_category
| limit 100
```

Sample output table results:

PATTERN	_TIME	APP_CATEGORY
true	Sep 3th 2024 09:13:37	general-ip
true	Sep 4th 2024 11:13:25	general-ip
true	Sep 6th 2024 15:22:36	multicast-ip

## 7 | Troubleshoot

### Abstract

Learn how to troubleshoot Cortex Agentix, such as sending audit notifications to a Syslog Server.

Troubleshoot errors, view management audit logs, set up a Syslog Server, configure management audit notification forwarding, and submit a case to support.



## 7.1 | About health issues

### Abstract

Cortex AgentiX provides health issues to help you monitor the health and integrity of supported Cortex AgentiX resources. Health issues comprise ingestion, collection, correlation, and event forwarding errors.

#### PREREQUISITE:

For Cortex AgentiX to monitor data ingestion health and create health issues, you must enable the following settings under Configurations:

Cortex - Analytics: Go to Configurations → Cortex - Analytics. For more information, see [Enable the Analytics Engine and Identity Analytics](#).

Cortex AgentiX provides health issues to help you monitor the health and integrity of supported Cortex AgentiX resources. Health issues provide insights into health drifts, such as failure events or status changes. The issues help you stay on top of your health related errors and ensure optimal performance in Cortex AgentiX. In addition, you can set up notifications on health issues.

Health issues are associated with the Health Domain. When setting up notification forwarding or other configurations for health issues, use the filter Issue Domain = Health.

To view health issues, go to Settings → Health Issues, or on the Issues page select the Health Domain table view. Click an issue to see more details in the issue card, or right-click to take actions and investigate an issue. For more information, see [Investigate and resolve health issues](#).

#### NOTE:

The Health Issues page displays issues that were triggered after July 2024. To see health issues that were triggered before this date, click [Legacy Health Issues](#).

### Types of health issues

Cortex AgentiX provides the following types of OOTB health issues:

- Ingestion issues:** Triggered by interruptions in data ingestion, or deviation from the calculated ingestion baseline
- Collection issues:** Triggered by connectivity errors in your collection integrations, custom collectors, and Marketplace integrations
- Correlation issues:** Triggered by correlation rules that complete with an error status
- Automation issues:** Triggered by system monitoring of metrics and thresholds for potential automation misconfigurations that can cause performance issues. Automation issues are processed daily to provide an aggregated status of multiple threshold crossings.

#### NOTE:

Cortex AgentiX enforces the dedup logic to health issues. This logic reduces the likelihood of identical health issues from flooding the issues dataset.

### Query health issue data

Health issues are associated with the Health domain. To query health issue data, use the following XQL:

```
dataset = alerts | filter alert_domain = "DOMAIN_HEALTH"
```

### Health issue field descriptions

The following table describes the health issue fields.

Field	Description
Issue ID	A unique identifier that Cortex AgentiX assigns to each issue.
Issue Name	Name of the issue.
Issue Type	Type of health issue.
Issue Source	Source of the issue.
Collector Name	Name of the collector instance.



Field	Description
Collector Type	Type of the collector.
Description	Text summary of the event including the issue source, issue name, and severity.
Device ID	Firewall device ID.
Excluded	Whether the issue is excluded.
External ID	Issue ID as recorded in the detector from which this issue was sent.
Final Reporting Device IP	IP of the device from which the log was extracted.
Final Reporting Device Name	Hostname of the device from which the log was extracted.
Ingestion Failure Duration	Amount of time that logs were not received or a drop in log ingestion was detected in minutes.
Observation Time	Time that the issue was observed in the system.
Playbook	Playbook that was run.
Playbook run status	Status of the playbook.
Product	Product name of the observing data source.
Resolution Status	Status that was assigned to this issue when it was triggered (or modified). Right-click an issue to change the status. If you set the status to Resolved, select a resolution reason.
Reporting Device Name	Host name of the device where the log originated.
Reporting Device IP	IP Address of the device where the log originated.
Severity	Severity level that was assigned to this issue when it was triggered (or modified).
Starred	Whether the issue is starred by starring configuration.
Vendor	Vendor of the observing data source.

#### 7.1.1 | Monitor data ingestion health

##### Abstract

Learn more about data ingestion health monitoring.



Cortex AgentiX collects granular data ingestion metrics that provide an insight into the data ingestion pipeline, and identify disruptions in data collection. With these metrics you can trace data collection from a specific source, and see a breakdown by data source attributes such as Collector Name and Final Reporting Device.

You can use these metrics in Cortex Query Language (XQL) queries to investigate disruption and degradation in log collection. You can also create correlation rules that use your own data ingestion logic to trigger issues when disruption occurs for a specific data source within a specific timeframe.

In addition, Cortex AgentiX has a built-in data ingestion monitoring and issues mechanism that monitors the availability and overall health of data ingestion in your environment, and triggers ingestion health issues if disruptions occur.

#### Related topics

- Overview of data ingestion metrics
- Creating correlation rules to monitor data ingestion health
- Measuring data freshness
- About health issues

### 7.1.2 | Monitor correlation rules

#### Abstract

You can monitor your correlation executions with the `correlations_auditing` dataset.

Cortex AgentiX audits all correlation executions in the `correlations_auditing` dataset. The dataset records the query initiation times, end times, retry attempts, failure reasons, and other useful metrics. You can use this dataset to monitor your correlation executions. Cortex AgentiX also provides OOTB health issues that are generated when a correlation rule completes with errors. For more information, see About health issues.

In the `correlations_auditing` dataset, audit entries are added as follows:

- The rule starts executing. This is audited with the status of Initiated or Initiated Manually.
- The rule completes successfully. This is audited as Completed.
- The rule completes with errors. This is audited as Error.

#### NOTE:

In the dataset, the Query start time and Query end time indicate the timeframe of the data that was queried. The actual start and end times of the correlation rule execution are recorded in the `_time` field for the Initiated and Completed entries.

#### Field descriptions for the correlations\_auditing dataset

The following table describes the fields in the `correlations_auditing` dataset:

Field	Description
<code>_time</code>	Timestamp of the audit. For entries with an Initiated or Initiated Manually status, this is the start time of the correlation rule execution. For entries with a Completed or Error status, this is the end time of the rule execution.
<code>_id</code>	Unique identifier of the audit entry.
Rule ID	Unique identification number for the correlation rule.
Name	Correlation rule name.
Status	The status of the correlation rule query. Possible values are Initiated, Initiated Manually, Completed, and Error.
Query start time	The start time of the query timeframe.



Field	Description
Query end time	The end time of the query timeframe.
Time frame	Time frame for the query.
Failure reason	For correlation rules with errors, this field displays the error message.
Retry attempts	Number of retry attempts before the query initiated or failed to run.
Schedule	Scheduled frequency to execute the correlation rule.
Rule creation time	Date and time that the correlation rule was created.
Rule modification time	Date and time that the correlation rule was last modified.
Description	Description of the correlation rule.
Severity	Defined severity of the correlation rule.
Dataset	Target data set, as defined in the correlation rule
Suppression status	Whether issue suppression is Enabled or Disabled.
Suppression duration	Duration for which to ignore additional events that match the issue suppression criteria.
Suppression fields	Fields on which the issue suppression is based.
Timezone	Timezone on which the scheduled frequency is based.
MITRE ATT&CK Tactic	MITRE ATT&CK tactic that the correlation rule attempted to generate.
MITRE ATT&CK Technique	MITRE ATT&CK technique that the correlation rule attempted to generate.
Issue category	Category of issue as configured when creating the rule.
Source	Source of the correlation rule.
XQL search	XQL query for the correlation rule.
Drill-down query	XQL query configured for further investigation.
Issue name	Name of the issue that the correlation rule will generate.



### 7.1.3 | Investigate and resolve health issues

#### Abstract

You can investigate and take action on health issues from the Health Issues page and the Issues Table.

The following tasks explain how to investigate and resolve health issues. You can see health issues on the following pages:

- Go to Settings → Health Issues
- Go to Cases & Issues → Issues and change the table view to Health Domain.

#### Investigate data ingestion errors

A data ingestion issue identifies disruption in the data ingestion pipeline. For example, a data source is not sending logs, or there is a significant drop in log collection compared to the calculated ingestion baseline.

1. Identify the error: Type = Ingestion.
2. Right-click and select Investigate in XQL query.

The Query Builder opens and runs a prefilled query to display related data ingestion metrics entries.

3. Review the query results.

The results provide context for the issue and the events leading up to it. For more information about data ingestion metrics and setting up correlation rules with your own data ingestion logic, see Monitor data ingestion health.

4. Investigate data collector errors. Return to the Health Issues page, right-click the issue, and select Pivot to views → View collector details.

Depending on the type of collector in error, the relevant data collector settings page opens, filtered by data collector.

#### Investigate collection errors

A collection issue identifies connectivity disruption in your collection integrations, custom collectors, and Marketplace integrations.

1. Identify the error: Type = Collection.
2. See the current status of the collector.

Right-click and select Pivot to views → View collector details. Depending on the type of collector in error, the relevant data collector settings page opens, filtered by data collector.

If the data collector is still in error, you can update the collector settings as required.

3. Investigate the collector error status.

Run a query on the `collection_auditing` dataset to see all the connectivity changes of the collector over time, the escalation or recovery of the connectivity status, and the error, warning, and informational messages related to status changes.

Example 121.

This example searches for status changes for the "instance1" data collector integration:

```
dataset = collection_auditing  
|filter collector_type = "STRATA_IOT" and instance = "instance1"
```

For more information about troubleshooting collector errors and setting up correlation rules to trigger additional collection issues, see Verify collector connectivity.

#### Investigate correlation errors

A correlation issue identifies errors in your correlation rules.

1. Identify the error: Type = Correlation.
2. Right-click and select Investigate Correlation Auditing.

The Query Builder opens and runs a prefilled query to display related correlation execution records.

3. Review the query results.

Identify the correlation rule in error and take steps to resolve the error. For more information about how Cortex AgentX identifies correlation rule errors, see Monitor correlation rules.



## Investigate automation errors

Automation issues identify potential misconfigurations in automations, enabling you to take a proactive approach to fixing misconfiguration issues before they affect system performance.

1. Identify the error: Type = Automation.
2. Click the automation health issue to view the details of the related case or component.
3. Based on the details of the automation health issue, review any related automations, such as playbooks and integrations, for possible misconfigurations.

## 7.2 | Log forwarding

### Abstract

Stay informed and updated about events in your system by forwarding alerts and reports to an external service, such as a syslog receiver, a Slack channel, or an email account.

Logs provide information about events that occur in the system. These logs are a valuable tool in troubleshooting issues that might arise in your Cortex AgentiX tenant.

To stay informed about important alerts and events, you can configure your notifications and specify the type of logs you want to forward. You can choose to receive these notifications through an email account, a Slack channel, or a syslog receiver.

### 7.2.1 | Forward logs from Cortex AgentiX to external services

#### Abstract

Learn how to forward logs from Cortex AgentiX to external services such as email, Slack, or a syslog receiver.

You can forward logs from Cortex AgentiX to an external service. This allows you to stay updated on important issues and events. Available services include the following:

- **Slack channel and/or syslog receiver:** Integrate the service with Cortex AgentiX. Once the integration is complete, configure notification forwarding, specifying the log type you want to forward.
- **Email distribution list:** Configure notification forwarding, specifying the log type you want to forward.

The following table shows the log types supported for each notification type:

Log Type	Email	Slack	Syslog
Issues	â	â	â
Cases	â	â	â
Management Audit Log	â	â€	â

#### 7.2.1.1 | Configure external applications for forwarding

#### Abstract

Configure external applications so you can forward data to services such as syslog servers, Slack, Splunk, Amazon SQS, Amazon S3, and Webhook.

Data and logs can be forwarded to third-party external services. The external service must be configured in Cortex AgentiX before you set up notification forwarding.

Before forwarding cases or issues to Splunk, Amazon S3, Amazon SQS, or Webhook, you need to configure egress in the Cortex Gateway. You do not need to configure egress for email, Slack, or syslog forwarding.

#### NOTE:



- No prior configuration is required to send data or logs to an email distribution list.
- Only cases and issues can be forwarded to Slack, Amazon S3, Amazon SQS, Splunk, and Webhook.
- To forward data to Amazon SQS, the following permissions are required: `sqs:GetQueueAttributes`, `sqs>ListQueues`, `sqs:SendMessage`, `sqs:SendMessageBatch`, `tag:GetResources`, and `iam:GetRole`.
- To forward data to Amazon S3, the following permissions are required: `s3:PutMessage`, `s3:PutObject`, and `s3>ListBucket`.

#### Task 1: Configure egress in Cortex Gateway

To forward data to Amazon S3, Amazon SQS, Splunk, or Webhook, you must first configure egress in Cortex Gateway. You do not need to configure egress to forward to Slack, email, or a syslog server.

##### **NOTE:**

Only a user with Account Admin or Instance Admin permissions can configure egress. For more information, see Egress configurations.

1. In the Cortex Gateway, go to Permission Management → Egress Configurations → Path.
2. Select the account name and tenant.
3. In the Flow field, select the third-party service you want to forward to.

##### **NOTE:**

For Amazon SQS or Amazon S3, you need to provide the queue/bucket name. For Webhook or Splunk, you need to provide the domain, including any subdomain.

4. Add the configuration.

#### Task 2: Configure access in your firewall

If you are forwarding to Splunk or Webhook, add the IP addresses for your tenant region to your firewall. For more information, refer to the list of ingress IPs in Enable access to required PANW resources.

#### Task 3: Configure external application

##### **NOTE:**

You can also configure external applications when you create a new forwarding configuration. After defining the configuration and setting the scope, you can Add Application and follow the instructions below for any of these destinations.

1. Go to Settings → Configurations → Integrations → External Applications → Add Application.
2. Choose one of the following applications:

##### Amazon S3

1. Enter the S3 URI.
2. Click Verify.
3. After verification is successful, you can enter the instance name and optional description.
4. Enter the Role ARN (Amazon Resource Name).
5. Enter the AWS region. For example, `eu-central-1`.
6. (Optional) Select the maximum duration to collect data before writing to a new file. The default is one hour.
7. Test the connection.

##### Amazon SQS

1. Enter the Queue URL.
2. Click Verify.
3. After verification is successful, you can enter the name and optional description.
4. Select either IAM Role or IAM Access Keys.
  - For IAM Role, enter the Role ARN (Amazon Resource Name).
  - For IAM Access Keys, enter the Access Key and Secret Key.



5. Test the connection.

#### Slack

For Slack, see Integrate Slack for outbound notifications.

#### Syslog

For syslog server configuration, see Integrate a syslog receiver.

#### Splunk

1. Enter the Splunk HTTP event collector URL. The URL can include a port. The connection must be HTTPS.

2. Click Verify.

#### **NOTE:**

If egress has not been configured in the Cortex Gateway, verification will fail and a message will display that the endpoint does not match any approved routes.

3. After verification is successful, you can enter the instance name and optional description.

4. Enter the authentication token for secure access to your Splunk instance.

5. Test the connection.

#### Webhook

1. Enter the Webhook URL. The URL can include a port. The connection must be HTTPS.

2. Click Verify.

#### **NOTE:**

If egress has not been configured in the Cortex Gateway, verification will fail and a message will display that the endpoint does not match any approved routes.

3. After verification is successful, you can enter the instance name and optional description.

4. (Optional) Show advanced settings to add HTTP headers.

5. Test the connection.

3. Click Connect.

#### Task 4: Configure notification forwarding

Configure notification forwarding to email or external services. For more information, see Configure notification forwarding.

##### 7.2.1.1.1 | [Integrate a syslog receiver](#)

#### Abstract

Define syslog settings and then configure notification forwarding to receive notifications about issues and reports.

A syslog receiver can be a physical or virtual server, a SaaS solution, or any service that accepts syslog messages.

To send Cortex AgentIX notifications to your syslog receiver, you first need to define the settings for the syslog receiver. After this is complete, you can configure notification forwarding.

#### Enable access

Before you begin, enable access to the following Cortex AgentIX IP addresses for your region in your firewall.

#### Americas

Region	Log Forwarding Address
United States - Americas (US)	35.232.87.9, 35.224.66.220
United States - Government	104.198.222.185, 35.239.59.210
Brazil (BR)	35.247.234.13, 34.39.178.116



<b>Region</b>	<b>Log Forwarding Address</b>
Canada (CA)	35.203.54.204, 35.203.52.255

EMEA (Europe, Middle East, Africa)

<b>Region</b>	<b>Log Forwarding IP Addresses</b>
France (FA)	34.163.100.253, 34.155.72.149
Germany (DE)	35.234.95.96, 35.246.192.146
Israel (IL)	34.165.194.4, 34.165.101.105
Italy (IT)	34.154.0.173, 34.154.71.94
Netherlands - Europe (EU)	34.90.202.186, 34.90.105.250
Poland (PL)	34.118.45.145, 34.118.126.170
Qatar (QT)	34.18.48.182, 34.18.43.40
Saudi Arabia (SA)	34.166.50.215, 34.166.55.72
South Africa (ZA)	34.35.70.253, 34.35.10.167
Spain (ES)	34.175.83.90, 34.175.230.150
Switzerland (CH)	34.65.228.95, 34.65.74.83
United Kingdom (UK)	34.105.227.105, 34.105.149.197

JPAC (Asia-Pacific)

<b>Region</b>	<b>Log Forwarding IP Addresses</b>
Australia (AU)	35.189.38.167, 34.87.219.39
Delhi (DL)	34.126.223.198, 34.131.110.15
India (IN)	34.93.247.41, 34.93.183.131
Indonesia (ID)	34.101.248.99, 34.101.176.232



Region	Log Forwarding IP Addresses
Japan (JP)	34.84.88.183, 35.243.76.189
Singapore (SG)	35.240.192.37, 34.87.125.227
South Korea (KR)	34.64.198.58, 34.47.86.20
Taiwan (TW)	35.234.2.208, 35.185.171.91

How to send issues or logs to a syslog receiver

1. Go to Settings → Configurations → Integrations → External Applications → Add Application and select Syslog.

2. Define the following parameters:

Parameter	Description
Name	Unique name for the server profile.
Destination	IP address or fully qualified domain name (FQDN) of the syslog receiver.
Port	Port number to send syslog messages.
Facility	Select one of the syslog standard values. The value maps to how your syslog server uses the facility field to manage messages. For details on the facility field, see RFC 5424.
Protocol	<p>Method of communication with the syslog receiver:</p> <ul style="list-style-type: none"> <li>• TCP: No validation is made on the connection with the syslog receiver. However, if an error occurred with the domain used to make the connection, the Test connection will fail.</li> <li>• UDP: No error checking, error correction, or acknowledgment. No validation is done for the connection or when sending data.</li> <li>• TCP + SSL: Cortex AgentiX validates the syslog receiver certificate and uses the certificate signature and public key to encrypt the data sent over the connection.</li> </ul>
Certificate	<p>The communication between Cortex AgentiX and the syslog destination can use TLS. In this case, upon connection, Cortex AgentiX validates that the syslog receiver has a certificate signed by either a trusted root CA or a self-signed certificate. You may need to merge the Root and Intermediate certificate if you receive a certificate error when using a public certificate.</p> <p>If your syslog receiver uses a self-signed CA, upload your self-signed syslog receiver CA. If you only use a trusted root CA leave the certificate field empty.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>• Up to TLS 1.3 is supported.</li> <li>• Verify the self-signed CA includes your public key.</li> </ul> <p>You can ignore certificate errors. For security reasons, this is not recommended. If you choose this option, data and logs will be forwarded even if the certificate contains errors.</p>

3. Test the parameters to ensure a valid connection, and click Connect when ready.

You can define up to five syslog receivers. Upon success, the table displays the syslog servers and their status.

After you integrate with your syslog receiver, configure your forwarding settings. For more information see, Configure notification forwarding.



#### Syslog receiver test message errors

When configuring a syslog message, Cortex AgentiX sends a test message. If a test message cannot be sent, Cortex AgentiX displays an error message to help you troubleshoot.

The following table includes descriptions and suggested solutions for the error messages:

Error Message	Description
Host Resolving Failed	The IP address or hostname you provided doesn't exist, or can't be resolved. Ensure you have the correct IP address or the hostname.
Configured Local Address	The IP address or hostname you provided is internal and can't be used. Ensure you have the correct IP address or the hostname.
Wrong Certificate Format	The certificate you uploaded is in an unexpected format and can't be used. The certificate must be an ASCII string or a bytes-like object. Re-create the certificate in the correct format, for example: -----BEGIN CERTIFICATE----- MIIDHTCCAgWgAwIBAgIQSwieRyGdh6BNRQyp406bnTANBgkqhkiG9w0BAQsFADAhMR8wHQYDVQQDEzTVVJTLUNoYXJsaWVBbHBoYS1Sb290MB4XDTHwMDQzMDE4MjEzNFO -----END CERTIFICATE-----
Connection Timed Out	Cortex AgentiX didn't connect to the syslog receiver in the expected time. This could be because your firewall blocked the connection or because the configuration of the syslog server caused it to drop the connection. Check the firewall logs and the connection using Wireshark.
Connection Refused	The syslog receiver refused the connection. This could be because your firewall blocked the connection or because the configuration of the syslog server caused it to drop the connection. Check the firewall logs and the connection using Wireshark.



Error Message	Description	
Connection Reset	<p>The connection was reset by the syslog receiver. This could be because your firewall blocked the connection or because the configuration of the syslog receiver caused it to drop the connection.</p>	<p>Check the firewall logs and the connection using Wireshark.</p>
Certificate Verification Failed	<ul style="list-style-type: none"> <li>The uploaded certificate couldn't be verified for one of the following reasons.             <ul style="list-style-type: none"> <li>The certificate doesn't correspond to the certificate on the syslog receiver and cannot be validated.</li> <li>The certificate doesn't have the correct hostname.</li> <li>You are using a certificate chain and didn't merge the certificates into one certificate.</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>Incorrect certificate: to check that the certificate you are uploading corresponds to the server syslog certificate, use the following OpenSSL command.             <pre>openssl verify -verbose -CAfile cortex_upload_certificate syslog_certificate</pre>             If the certificate is correct, the result is <b>syslog_certificate: OK</b>.         </li> <li>Incorrect hostname: make sure that the hostname/ip in the certificate matches the syslog server.</li> <li>Certificate chain: If you are using a list of certificates, merge the chain into one certificate. You can concatenate the certificates.             <pre>cat intermediate_cert root_cert &gt; merged_syslog.crt</pre>             If the concatenated certificate doesn't work, change the order of the root and intermediate certificates, and try again.             To verify that the chain certificate was saved correctly, use the following OpenSSL command.             <pre>openssl verify -verbose -CAfile cortex_upload_certificate syslog_certificate</pre>             If the certificate is correct, the result is <b>syslog_certificate: OK</b>.         </li> </ul>



Error Message	Description
Connection Terminated Abruptly	The firewall or the syslog receiver dropped the connection unexpectedly. This could be because the firewall on the customer side limits the number of connections, the configuration on the syslog receiver drops the connection, or the network is unstable.
Host Unreachable	The network configuration is faulty, and the connection can't reach the syslog receiver.
SSL Error	Unknown SSL error.
Connection Unavailable	General error.

#### 7.2.1.1.2 | Integrate Slack for outbound notifications

##### Abstract

Learn how to integrate Cortex AgentiX with your Slack workspace and stay updated on important alerts and events.

Integrate Cortex AgentiX with your Slack workspace to manage and highlight your issues and reports. Creating a Cortex AgentiX Slack channel ensures that defined issues are exposed on laptop and mobile devices using the Slack interface. Unlike email notifications, Slack channels provide dedicated spaces where you can contact specific members regarding your issues.

##### How to integrate Slack with Cortex AgentiX

1. Go to Settings → Configurations → Integrations → External Applications → Add Application and click Slack.
2. Click Ok to go to an external Slack page to install Cortex AgentiX on your Slack workspace.

##### **NOTE:**

You are directed to the Slack browser to install Cortex AgentiX. You can only use this link to install Cortex AgentiX on Slack. Attempting to install from Slack Marketplace will redirect you to Cortex AgentiX documentation.

3. Click Submit.

Upon successful installation, Cortex AgentiX displays the workspace to which you connected.

##### What to do next

After you integrate with your Slack workspace, configure your forwarding settings. For more information see, Configure notification forwarding. To send reports to Slack, see Run or schedule reports.



## 7.2.1.2 | Configure notification forwarding

### Abstract

Learn how to create a forwarding configuration that specifies the log type you want to forward.

After you integrate with an external service such as Slack or a syslog receiver, create a forwarding configuration that specifies the log type you want to forward. You can configure notifications for issues, agent audit logs, and management audit logs. To receive notifications about reports, see Run or schedule reports.

#### PREREQUISITE:

Before you can select a syslog receiver or a Slack channel, you must integrate these external services with Cortex AgentiX.

For more information, see:

- [Integrate a syslog receiver](#)
- [Integrate Slack for outbound notifications](#)

### How to configure notifications

1. Select Settings → Configurations → General → Notifications.

2. Click + Add Forwarding Configuration.

3. Enter a name and description for the configuration.

4. Select the log type you want to forward:

- Issues: Send notifications for specific issue types.

#### NOTE:

- **Notification Forwarding by Domain:** To configure notification forwarding for issues by domain, select Log Type = Issues and filter the Issues table by Issue Domain.
- **Case IDs in Notifications:** If case matching completes before the notification is sent, the Case ID is included. If matching takes longer than the notification trigger, the notification is sent without a Case ID to prioritize timely visibility in the target system.
- **Alert vs. Issue Format:** New alert notifications are created using the notification forwarding configuration. By default, new configurations use the Issue format, but you can select the Alert format if needed.

Existing configurations are not updated automatically and will continue to send notifications in the Alert format. To use the Issue format, edit the existing configuration.

- Agent Audit Logs: Send notifications for audit logs reported by your Cortex XDR agents.

- Management Audit Logs: Send notifications for audit logs about events related to your Cortex AgentiX tenant.

- Health Issues: Send notifications for health issues. (for example, Ingestion, Event Forwarding, and Correlation rule errors).

#### NOTE:

This option will be deprecated in the next release. Configure issues with the filter Issue Domain = Health instead.

- Cases: Send notifications for specific cases (for example, Security cases)

5. Click **Next**, and under **Scope**, filter the type of information you want included in a notification.

For example, for a filter set to **Severity = Medium, Category = Configuration**, Cortex AgentiX sends the issues or events matching this filter as a notification.

6. Click **Next**.

7. (Optional) Define your email configuration:

- a. In the Distribution List, add the email addresses to which you want to send email notifications.

- b. In the Grouping Timeframe, define the time frame, in minutes, to specify how often Cortex AgentiX sends notifications. Every 20 issues or 20 events aggregated within this time frame are sent together in one notification, sorted according to severity. To send a notification when one issue or event is generated, set the time frame to 0.

- c. Choose whether you want Cortex AgentiX to provide an auto-generated subject.

8. Depending on the notification integrations supported by the log type, configure the Slack channel or syslog receiver notification settings. For a list of log types supported in each notification type, see [Forward logs and data from Cortex AgentiX to external services](#).

1. Enter the Slack channel name and select from the list of available channels. Slack channels are managed independently of Cortex AgentiX in your Slack workspace. After integrating your Slack account with your Cortex AgentiX tenant, Cortex AgentiX displays a list of specific Slack channels associated with the integrated Slack workspace.

2. Select a syslog receiver. Cortex AgentiX displays the list of receivers integrated with your Cortex AgentiX tenant.



9. Click Done to create the forwarding configuration.

#### 7.2.1.3 | Monitor administrative activity

##### Abstract

View all Cortex AgentiX administrator-initiated actions taken on issues, cases, and live terminal sessions.

From Settings → Management Audit Logs, you can track the status of all administrative and investigative actions. Cortex AgentiX stores audit logs for 365 days. Use the page filters to narrow the results or manage tables to add or remove fields as needed.

To ensure you and your colleagues stay informed about administrative activity, you can configure notification forwarding to forward your Management Audit log to an email distribution list, Syslog server, or Slack channel.

The following table describes the default **and optional fields** that you can view in alphabetical order.

Field	Description
Email	Email address of the administrative user
Description	Descriptive summary of the administrative action. Hover over this field to view more detailed information in a pop-up tooltip. This enables you to know exactly what has changed, and, if necessary, roll back the change.
Host Name	Name of any relevant affected hosts
ID	Unique ID of the action
Result	Result of the administrative action: Success, Partial, or Fail.
Subtype	Subcategory of action
Timestamp	Time and date of the action



Field	Description
Type	<p>Type of activity logged, one of the following:</p> <ul style="list-style-type: none"> <li>• Issue Exclusions: Suppression of particular issues from Cortex AgentiX .</li> <li>• Issue Fields: Modification of issue fields.</li> <li>• Issue Layouts: Modification of issue layouts.</li> <li>• Issue Layout Rules: Modification of issue layout rules.</li> <li>• Issue Notifications: Modification of the format or timing of issues.</li> <li>• Issue Rules: Modification of issue rules.</li> <li>• API Key: Modification of the Cortex AgentiX API key.</li> <li>• Authentication: User sessions started, along with the user name that started the session.</li> <li>• Dashboards: Use of particular dashboards.</li> <li>• Case Management: Actions taken on cases, issues, and artifacts in cases.</li> <li>• Ingest Data: Import of data for immediate use or storage in a database.</li> <li>• Integrations: Integration operations, such as integrating Slack for outbound notifications.</li> <li>• Licensing: Any licensing-related operation.</li> <li>• Managed Threat Hunting: Activity relating to managed threat hunting.</li> <li>• MSSP: Management of security services providers.</li> <li>• Public API: Authentication activity using an associated Cortex AgentiX API key.</li> <li>• Query Center: Operations in the Query Center.</li> <li>• Remediation: Remediation operations.</li> <li>• Reporting: Any reporting activity.</li> <li>• Response: Remedial actions taken. For example: Isolate a host, undo host isolation, add a file hash signature to the block list, or undo the addition to the block list.</li> <li>• Rules: Modification of rules.</li> <li>• Rules Exceptions: Creation, editing, or deletion under Rules exceptions.</li> <li>• SaaS Collection: Any collected SaaS data.</li> <li>• Script Execution: Any script execution.</li> <li>• Starred Cases: Modification of starred cases.</li> <li>• Vulnerability Assessment: Any vulnerability assessment activity.</li> </ul>
User Name	The user who performed the action.

### 7.2.2.1 Data and log notification formats

#### Abstract

Cortex AgentiX provides you with different formats for its log notifications.

When Cortex AgentiX cases, issues, and logs are forwarded to email or a third-party system, notifications are sent in a specific format.

**NOTE:**



Issues can be forwarded to email, syslog servers, and Slack in the alert format, if you prefer. The alert format can be selected when you configure your forwarding notification.

### 7.2.2.1 | Issue notification format

#### Abstract

Learn about the formats used to forward correlation and third-party issues.

Correlation and third-party issues are forwarded to external data resources according to the email, Slack, or syslog format.

#### Email account

Cortex AgentIX sends issue notifications to email accounts based on the settings you configure. Email messages also include an issue code snippet of the fields according to the columns in the Issue table.

The notification format is as follows:

- If only one issue exists in the queue, a single-issue email format is sent.
- If more than one issue was grouped in the time frame, all the issues in the queue are forwarded together in a grouped email format.

Example 122.

Single-issue email message

```
Email Subject: Issue: <issue_name>
Email Body:
  Issue Name: Suspicious Process Creation
  Severity: High
  Source: Correlation
  Category: Malware
  Action: Detected
  Host: <host name>
  Username:<user name>
  Excluded: No
  Starred: Yes
  Issue: <link to the tenant issue view>
  Case: <link to the tenant case view>
```

Example 123.

Grouped issue email message

```
Email Subject: Issues: <first_highest_severity_issue> + x others
Email Body:
  Issue Name: Suspicious Process Creation
  Severity: High
  Source: Correlation
  Category: MalwareAction: Detected
  Host: <host name>
  Username:<user name>
  Excluded: No
  Starred: Yes
  Issue: <link to the tenant issue view>
  Case: <link to the tenant case view>
  Issue Name: Behavioral Threat Protection
  Issue ID: 2412
  Description: A really cool detection
  Severity: Medium
  Source: Correlation
  Category: Exploit
  Action: Prevented
  Host: <host name>
  Starred: Yes
  Case: <link to the tenant case view>
  Issue: <link to the tenant issue view>
  Notification Name: "My notification policy 2"
  Notification Description: "Starred issues with medium severity"
```

Example 124.

Email body

```
{
  "original_issue_json": {
    "uid": "<UUID Value>",
    "recordType": "threat",
    "customerId": "<Customer ID>",
    "severity": 4,
    "...",
    "is_pcap": null,
    "contains_featured_host": [
      "NO"
    ],
    ...
  }
}
```



```

"contains_featured_user": [
    "YES"
],
"contains_featured_ip": [
    "YES"
],
"events_length": 1,
"is_excluded": false
}

```

#### Slack channel

You can send issue notifications to a single Slack contact or a Slack channel. Notifications are similar to the email format.

#### Syslog receiver

Issue notifications forwarded to a syslog receiver are sent in a CEF format RF 5425.

Section	Description
Syslog header	<9>: PRI (considered a priority field)1: version number2020-03-22T07:55:07.964311Z: timestamp of when alert/log was sentcortexxdr: host name
CEF header	HEADER/Vendor="Palo Alto Networks" (as a constant string)HEADER/Device Product="Cortex XDR" (as a constant string)HEADER/Product Version= Cortex XDR version (2.0/2.1....)HEADER/Severity=(integer/0 - Unknown, 6 - Low, 8 - Medium, 9 - High)HEADER/Device Event Class ID=alert sourceHEADER/name =alert name
CEF body	end=timestamp shost=endpoint_name deviceFacility=facility cat=category externalId=external_id request=request cs1=initiated_by_process cs1Label=Initiated by (constant string) cs2=initiator_commande cs2Label=Initiator CMD (constant string) cs3=signature cs3Label=Signature (constant string) cs4=cgo_name cs4Label=CGO name (constant string) cs5=cgo_command cs5Label=CGO CMD (constant string) cs6=cgo_signature cs6Label=CGO Signature (constant string) dst=destination_ip dpt=destination_port src=source_ip spt=source_port fileHash=file_hash filePath=file_path targetprocesssignature=target_process_signature tenantname=tenant_name tenantCDLid=tenant_id CSPaccountname=account_name initiatorSha256=initiator_hash initiatorPath=initiator_path osParentName=parent_name osParentCmd=parent_command osParentSha256=parent_hash osParentSignature=parent_signature osParentSigner=parent_signer incident=incident_id act=action suser=actor_effective_username

#### Example 125.

```

end=timestamp shost=endpoint_name deviceFacility=facility cat=category externalId=external_id request=request cs1=initiated_by_process cs1Label=Initiated by (constant string) cs2=initiator_commande cs2Label=Initiator CMD (constant string) cs3=signature cs3Label=Signature (constant string) cs4=cgo_name cs4Label=CGO name (constant string) cs5=cgo_command cs5Label=CGO CMD (constant string) cs6=cgo_signature cs6Label=CGO Signature (constant string) dst=destination_ip dpt=destination_port src=source_ip spt=source_port fileHash=file_hash filePath=file_path targetprocesssignature=target_process_signature tenantname=tenant_name tenantCDLid=tenant_id CSPaccountname=account_name initiatorSha256=initiator_hash initiatorPath=initiator_path osParentName=parent_name osParentCmd=parent_command osParentSha256=parent_hash osParentSignature=parent_signature osParentSigner=parent_signer incident=incident_id act=action suser=actor_effective_username

```



## 7.2.2.2 | Management Audit log notification format

### Abstract

An email account or a syslog receiver are the notification channels through which the Management Audit log is communicated.

Cortex AgentiX forwards the Management Audit log to these external data sources:

- **Email account:** Sent according to the settings you configured. For more information, see Configure notification forwarding.
- **Syslog receiver:** Sent in a CEF format RFC 5425 according to the following mapping:

Section	Description
Syslog header	<9>: PRI (considered a priority field)1: version number2020-03-22T07:55:07.964311Z: timestamp of when issue /log was sentcortexxdr: host name
CEF header	HEADER/Vendor="Palo Alto Networks" (as a constant string)HEADER/Device Product="Cortex XDR" (as a constant string)HEADER/Device Version= Cortex XDR version (2.0/2.1....)HEADER/HEADER/Severity=(integer/0 - Unknown, 6 - Low, 8 - Medium, 9 - High)HEADER/Device Event Class ID="Management Audit Logs" (as a constant string)HEADER/name = type
CEF body	suser=user end=timestamp externalId=external_id cs1Label=email (constant string) cs1=user_email cs2Label=subtype (constant string) cs2=subtype cs3Label=result (constant string) cs3=result cs4Label=reason (constant string) cs4=reason msg=event_description tenantname=tenant_name tenantCDLid=tenant_id CSPaccountname=csp_id

### Example 126.

```
3/18/2012:05:17.567 PM<14>1 2020-03-18T12:05:17.567590Z cortexxdr -- CEF:0|Palo Alto Networks|Cortex XDR|Cortex XDR x.x |Management Audit Logs|REPORTING|6|suser=test end=1584533117501 externalId=5820 cs1Label=email cs1=test@paloaltonetworks.com cs2Label=subtype cs2=Slack Report cs3Label=result cs3=SUCCESS cs4Label=reason cs4=None msg=Slack report 'scheduled_1584533112442' ID 00 to ['CUXM741BK', 'C01022YU00L', 'CV51Y1E2X', 'CRK3VASN9'] tenantname=test tenantCDLid=11111 CSPaccountname=00000
```

## 7.3 | In-product support case creation

### Abstract

Open a support case directly in Cortex AgentiX and record your console to capture your issues and have the case handled efficiently.

To simplify the process of creating a support case, you can open a support case directly in Cortex AgentiX. Opening the case in Cortex AgentiX allows all of the relevant context to be included, such as the option to record the console and upload relevant logs. When relevant, Cortex AgentiX will create and send the agent tech support file (TSF) for the endpoint you select. All relevant data about your tenant is logged and included in the support case, including license details. Using the Cortex AgentiX Create Support Case Wizard makes it easier for you to include all of the necessary details and log files while first submitting your support case, thereby enabling the support team to solve it more quickly and easily.

To use the embedded support case feature, you must have a user account in the Customer Support Portal, and your Cortex AgentiX user must be granted the Help permission in Cortex Gateway.

1. From Cortex AgentiX, select Help → Submit a Support Case.
2. In the Submit Support Case wizard, enter the requested case information. Be precise when indicating the impact of the issue. When an issue is critical, you will be asked to input the most critical information so that support can understand the issue and start addressing it immediately.

#### NOTE:

When opening a support case through the Customer Support Portal, you need to manually select Cortex AgentiX as the product. While there may be discrepancies between the categories in this wizard and the Customer Support Portal process, that's because this wizard is designed specifically to focus on options relevant to Cortex AgentiX.

3. When the issue you are opening a support case for is related to the agent, you can select the relevant endpoint. If you select the endpoint, Cortex AgentiX will create and send the TSF for the agent you selected, when possible.



**NOTE:**

Selecting an endpoint from the endpoint table and retrieving TSF requires full Retrieve Endpoint Data permissions under Endpoint Administration.

4. To provide more context for your support case, you can record the Cortex AgentiX console directly from the support case wizard. If you choose to record the console, you can also opt to have the HAR file generated and sent to further assist support in solving the case. To record the console, select Record Console. To submit your support case without recording the console, select Skip.

5. If you choose to record the console, your browser may prompt you for permission for Cortex AgentiX to see the contents of the tab. To allow recording, select Allow. You can now recreate the issue in your Cortex AgentiX environment and all of your actions are recorded. The console recording and HAR file generation only take place within the context of the browser tab that Cortex AgentiX is running in. When you are ready to stop recording, select Stop Sharing.

If you wish to recreate the recording, you must first delete the existing console recording by clicking the x symbol next to the Console Recording. Then select Record Console.

**NOTE:**

Console recordings cannot exceed 10 minutes. The current recording time is displayed at the top of the window.

6. To submit the support case, click Submit Support Case.

While the case attachments are uploading, do not refresh or navigate away from Cortex AgentiX until you get a notification in the Notification Center that uploading is complete. In the meantime, you can close this wizard and continue working in Cortex AgentiX.

Once the support case is created successfully, the support case number is displayed and you will receive an email notification from Palo Alto Networks Support. You can manage the support case and monitor its progress in the Customer Support Portal.

## 8 | Reference

### 8.1 | Cortex AgentiX API

#### Abstract

Generate an API key and make your first API call.

Before you can begin using Cortex AgentiX APIs, you must generate the following items from Cortex AgentiX:

Value	Description
API Key	<p>The API Key is your unique identifier used as the Authorization:{key} header required for authenticating API calls.</p> <p>Depending on your desired security level, you can generate two types of API keys, Advanced or Standard.</p> <p>The Advanced key hashes the key using a nonce, a random string, and a timestamp to prevent replay attacks. CURL does not support this but it can be used with scripts.</p>
API Key ID	<p>The API Key ID is your unique token used to authenticate the API Key. The header used when running an API call is <code>x-xdr-auth-id:{key_id}</code>.</p>
FQDN	<p>The FQDN is a unique host and domain name associated with each tenant. When you generate the API Key and Key ID, you are assigned an individual FQDN.</p>

Cortex AgentiX API URIs comprise of your unique FQDN, the API name, and the name of the call. For example, `https://api-{fqdn}/agentix/{name_of_api}/{name_of_call}/`.

The following steps describe how to generate the necessary key values.

#### Task 1. Create an API key

1. Select Settings â Configuration â Integrations â API Keys â New Key.
2. Select the type of API Key you want to generate based on your desired security level: Advanced or Standard.



3. To define a time limit on the API key authentication, mark Enable Expiration Date and select the expiration date and time.

You can view the Expiration Time field for each API key at Settings & Info â—> Settings â—> Integrations â—> API Keys . In addition, Cortex AgentIX displays an API Key Expiration notification in the Notification Center one week and one day before the defined expiration date.

4. (Optional) Provide a comment that describes the purpose of the API key.

5. Expand each area to select the desired level of access for this key.

Select a role from the list of existing Roles, or a Custom Role from the list of previously defined roles to set the permissions on a more granular level. You can select multiple roles.

6. Generate the API Key.

7. Copy the API key, and then click Close. This value represents your unique `Authorization:{key}`.

**CAUTION:**

You can't view the API Key again after you complete this step so ensure that you copy it before closing the notification.

## Task 2. Get your Cortex AgentIX API key ID

1. In the API Keys table, locate the ID field.

2. Note your corresponding ID number. This value represents the `x-xdr-auth-id:{key_id}` token.

## Task 3 Get your FQDN

1. Go to Settings â—> Configurations â—> Integrations â—> API Keys.

2. Click Copy API URL

Your FQDN is saved in the clipboard. You must use the FQDN as part of the URL in every API call. For example: `https://api-company.us.com/agentix/public/v1/{endpoint_path}/`

## Task 3 Make your first API call

The following examples vary depending on the type of key you select.

You can test authentication with Advanced API keys using the provided Python 3 example. With Standard API keys, use either the cURL example or the Python 3 example. Don't forget to replace the example variables with your unique API key, API key ID, and FQDN tenant ID.

After you verify authentication, you can begin making API calls.

Example 127. Create an incident - Standard key cURL example

```
curl -X POST https://api-company.us.com/agentix/public/v1/incident
-H "x-xdr-auth-id:{api_key_id}"
-H "Authorization:{api_key}"
-H "Content-Type:application/json"
--data '{
"details": "My test incident",
"name": "My test incident",
"severity": 2,
"type": "Unclassified"
}'
```

## 8.2 | XDM fields for mapping authentication events

### Abstract

Learn more about the Cortex Data Model (XDM) fields to map for authentication events.

This section provides a comprehensive guide to mapping authentication events from various customer log sources to the XDM (Cortex Data Model) schema. Each relevant XDM field is detailed, including whether the field is mandatory or optional, the corresponding Authentication Story field , data type, and purpose, ensuring consistent data normalization essential for robust security analysis and threat detection.

The fields that are mandatory to map are listed below with an asterisk (\*) beside them as these fields must be mapped to automatically create authentication stories for XDM identity data.

**NOTE:**

For more information on the entire Cortex Data model (XDM) schema, see Cortex XSIAM Data Model Schema.



1. xdm.source.port\*

Authentication Story Field: action\_local\_port

Type: integer

Requirement: Mandatory

Data Model Rule example:

```
xdm.source.port=to_integer(0)
```

2. xdm.target.ipv4\*

Authentication Story Field: action\_remote\_ip

Type: string

Requirement: Mandatory

Note: Map a field that isn't a String or list. If there is no relevant field to map to the target IP, populate this field with an empty string ("").

Data Model Rule example:

```
xdm.target.ipv4 = ""
```

3. xdm.target.port\*

Authentication Story Field: action\_remote\_port

Type: integer

Requirement: Mandatory

Data Model Rule example:

```
xdm.target.port=to_integer(0)
```

4. xdm.network.ip\_protocol\*

Authentication Story Field: action\_network\_protocol

Type: integer

Requirement: Mandatory

See full list of options for this enum in the XDM IP Protocol documentation.

Data Model Rule example:

```
xdm.network.ip_protocol=XDM_CONST.IP_PROTOCOL_TCP
```

5. xdm.event.type\*

Authentication Story Field: dfe\_labels

Type: string

Requirement: Mandatory

Description: Represents events related to authentication activity, such as login attempts, SSO sessions, and MFA challenges.

Required Value: Must contain "authentication"

Data Model Rule example:

```
xdm.event.type = if(eventType in ("user.authentication.sso", "user.session.start", "user.mfa.okta_verify.deny_push", "user.mfa.factor.update", "user.authentication.auth_via_mfa", "user.authentication.auth_via_AD_agent", "user.authentication.verify", "user.authentication.auth_via_radius", "user.authentication.auth_via_richclient", "system.push.send_factor_verify_push"), "authentication", "")
```

6. xdm.event.tags\*

Authentication Story Field: dfe\_labels

Type: array<string>

Requirement: Mandatory

Description: Represents events related to authentication activity, such as login attempts, SSO sessions, and MFA challenges.



**Required Value:** Must contain XDM\_CONST.EVENT\_TAG\_AUTHENTICATION

**Data Model Rule example:**

```
xdm.event.tags = arraycreate(XDM_CONST.EVENT_TAG_AUTHENTICATION)
```

7. xdm.source.ipv4\*

Authentication Story Field: action\_local\_ip

Type: string

Requirement: Mandatory

Description: Represents the external IPv4 address from which the user authenticated. This is the IP address observed by the identity provider or SaaS system when the authentication request was processed. It typically reflects the user's public-facing network location, such as their home IP, office gateway, or VPN egress point.

**NOTE:**

Do not map a static string, list, or empty string. You must map this field from the raw log field that best represents the actual source IP used in the authentication attempt. In cases where multiple IP fields are available, such as client\_ip, source\_ip, and original\_client\_ip, choose the field that captures the IP address from which the user initially triggered the authentication request - before any processing by proxies or intermediate systems.

8. xdm.event.operation\*

Authentication Story Field: event\_sub\_type

Type: string

Requirement: Mandatory

Description: This field describes the type of authentication flow, such as a regular login or a multi-factor authentication (MFA) event. It helps standardize different event types from various sources into two clear categories - making detection logic easier to build, understand, and reuse across systems.

Supported values:

- XDM\_CONST.OPERATION\_TYPE\_AUTH\_LOGIN: Login using only a password
- XDM\_CONST.OPERATION\_TYPE\_AUTH\_MFA: Login that involves multi-factor authentication

**Data Model Rule example:** The following pseudocode is an example only, which must be modified to implement the logic in the correct syntax by the mapped data source to different event types.

```
xdm.event.operation =
  if eventType IN ("authentication", "oauth2", "mfa", "mfa_challenge", "mfa_verify")
  then XDM_CONST.OPERATION_TYPE_AUTH_MFA
  else if eventType IN ("session", "access_request", "iwa", "ldap")
  then XDM_CONST.OPERATION_TYPE_AUTH_LOGIN
  else if is_null(eventType)
  then null else to_string(eventType)
```

9. xdm.event.original\_event\_type\*

Authentication Story Field: sso\_event\_type

Type: string

Requirement: Mandatory

Description: This field captures the original name or label of the event as it appears in the raw log source. It serves as a direct reflection of the source-specific event type and is essential for maintaining source context.

Example values:

- user.authentication.sso(Okta – SSO authentication event)
- user.mfa.okta\_verify.push.deny(Okta – MFA denied)
- user.session.start(Okta – Session initiated)
- microsoft.login.success(Azure AD – Successful login)
- microsoft.mfa.challenge.fail(Azure AD – MFA failure)
- google.sign\_in.challenge(Google Workspace – Sign-in challenged)
- user.lockout(Generic – Account locked due to failed attempts)



## 10. xdm.auth.service\*

Authentication Story Field: auth\_service

Type: string

Requirement: Mandatory

Description: This field defines the role the system played in the authentication flow, such as identity provider or relying party, and should reflect event-specific context.

Supported values:

- `SP` (Service Provider): The system initiating the authentication request.
- `IDP` (Identity Provider): The system that validates the user authentication.

Data Model Rule example for Okta:

```
if(eventType = "user.authentication.auth_via_AD_agent", "IDP",
  eventType = "user.authentication.auth_via_radius", "IDP", ..., eventType = "user.authentication.sso", "SP", null)
```

### NOTE:

Mapping should be done per event type. The same system could be an IDP in one event and an SP in another.

## 11. xdm.event.outcome\*

Authentication Story Field: auth\_outcome

Type: string (ENUM)

Requirement: Mandatory

Description: Specifies the final result of the authentication attempt. Use values such as OUTCOME\_SUCCESS or OUTCOME\_FAILURE only for events that definitively represent the conclusion of the authentication process, such as when access is explicitly granted or denied. Avoid assigning these values to intermediate steps that don't reflect the final outcome.

Supported values:

- XDM\_CONST.OUTCOME\_SUCCESS
- XDM\_CONST.OUTCOME\_FAILED

### NOTE:

For more information on the event outcome constants in the Cortex Data Model, see XDM\_CONST.OUTCOME.

Data Model Rule example logic:

```
if(res == "[ss]success" OR res = "sent",
  XDM_CONST.OUTCOME_SUCCESS, res == "fail",
  XDM_CONST.OUTCOME_FAILED)
```

### NOTE:

Outcome is based on a conclusive event type reflecting the true end state of the authentication flow. Critical for effectiveness of detection rules. Incorrect derivation can lead to missed detections or false positives.

## 12. xdm.event.outcome\_reason

Authentication Story Field: auth\_outcome\_reason\_category

Type: string

Requirement: Optional

Description: Specifies a standardized and descriptive reason for the outcome of an authentication or access-related event. This field offers detailed context beyond a simple success or failure result and is essential for accurate risk assessment, efficient triage, and effective incident response.

Supported values:



- user\_does\_not\_exist
- bad\_credentials
- account\_expired\_or\_disabled
- account\_locked
- failed\_login
- auth\_policy\_accessViolation
- mfa\_failure
- mfa\_expired
- user\_reject
- user\_cancelled
- OTHER
- NOT\_SPECIFIED

**Required action:** Explicit mapping logic is required between the raw event fields that contain outcome/error messages, such as `get_reason` and `debug_data_error_code`, and this canonical field. Mapping must normalize provider-specific strings or error codes into one of the supported values.

**Data Model Rule example:** The following pseudocode is an example only, which must be modified to implement the logic in the correct syntax.

```
xdm.event.outcome_reason = if( get_reason == "UNKNOWN_USER", "user_does_not_exist",
    get_reason == "Login denied. No matching user", "user_does_not_exist",
    get_reason == "INVALID_CREDENTIALS", "bad_credentials", debugdata_errorcode == "1326", "bad_credentials",
    get_reason == "account is expired", "account_expired_or_disabled",
    get_reason == "USER_ACCOUNT_EXPIRE", "account_expired_or_disabled", debugdata_errorcode IN ("1331", "1793"),
    "account_expired_or_disabled",
    get_reason == "account is locked", "account_locked",
    get_reason == "LOCKED_OUT", "account_locked",
    get_reason == "Login failed", "failed_login",
    get_reason == "PASSWORD_BASED_LOGIN_DISALLOWED", "auth_policy_accessViolation",
    get_reason == "login denied", "auth_policy_accessViolation",
    get_reason == "VERIFICATION_ERROR", "mfa_failure",
    get_reason == "DEL_AUTH_TIMEOUT", "mfa_expired",
    get_reason == "User rejected Okta push verify", "user_reject",
    get_reason == "Login aborted", "user_cancelled",
    get_reason == "NOT_SPECIFIED", "OTHER")
```

#### 13. xdm.source.user.upn\*

**Authentication Story Field:** auth\_identity

**Type:** string

**Requirement:** Mandatory

**Description:** Represents the user identity associated with the authentication or access event. This field must be populated using the User Principal Name (UPN) format. It cannot be left empty.

Using UPN as a normalized identity format ensures consistency across diverse identity providers, such as Azure AD, Okta, and on-prem AD, and authentication flows. It plays a central role in correlating activity across logs, enriching detections, and building an accurate authentication story across systems.

**Example values:** jane.doe@company.com

#### 14. xdm.event.description

**Authentication Story Field:** sso\_display\_message

**Type:** string

**Requirement:** Optional

**Description:** Provides a human-readable summary describing the nature of the authentication event. This value is typically derived from the source system's descriptive message and is intended to offer clear context for analysts during monitoring and investigations.

**Example values:**

- A push was sent to a user for verification
- "User single sign on to app"
- Authentication of user via MFA



## 15. xdm.source.host.device.id

Authentication Story Field: `agent_id`

Type: string

Requirement: Optional

Description: This field represents the unique identifier of the device that initiated or is associated with the current authentication event.

The value should remain consistent per device over time and be mapped from the most reliable source-specific field, such as device ID, machine ID, or equivalent. If there isn't a sufficient device ID field, the alternative will be to map the IP address that initiated the authentication (same as `xdm.source.ipv4`).

## 16. xdm.source.host.hostname

Authentication Story Field: `auth_device_name`

Type: string

Requirement: Optional

Description: Represents the host/device name associated that initiated or is responsible for authentication events.

## 17. xdm.logon.type

Authentication Story Field: `auth_is_interactive`

Type: string

Requirement: Optional

Description: Represents the type of logon associated with the authentication event, with a focus on distinguishing between interactive (user-driven) and non-interactive (system or service-based) activity. This distinction is important for behavioral analysis, risk scoring, and threat detection.

Common values:

- `XDM_CONST.LOGON_TYPE_INTERACTIVE`: Indicates a user is interactively using the machine, such as logging in through a terminal session, remote shell, or console.
- `XDM_CONST.LOGON_TYPE_SERVICE`: Indicates a service-type logon, such as Windows service, automations, and application tokens, where the account must have the service logon privilege.

### NOTE:

These are the most common values, but other logon types also exist. For a complete list, see `XDM_CONST.LOGON_TYPE`.

## 18. xdm.event.operation.sub\_type

Authentication Story Field: `auth_method`

Type: string

Requirement: Optional

Description: Specifies a standardized and descriptive reason for the outcome of an authentication or access-related event. This field offers detailed context beyond a simple success or failure result and is essential for accurate risk assessment, efficient triage, and effective incident response.

Required action: You must define explicit mapping logic between the raw field in the source log that contains the authentication method information, such as `authMethod`, `authenticationFlow.type`, and `factorUsed`, and this XDM field.

This mapping should translate raw values into one of the allowed sub-category listed below.

Data Model Rule example: The following pseudocode is an example only, which must be modified to implement the logic in the correct syntax.

```
if(lowercase_auth_method = "password",      "password",
   lowercase_auth_method = "otp_sms",        "sms",
   lowercase_auth_method = "push",           "application",
   lowercase_auth_method = "yubikey",         "hardware_token",
   lowercase_auth_method = "trusted_device", "trusted_login",
   lowercase_auth_method = "sso",             "Generic SSO",
   lowercase_auth_method = "email",           "email",
   lowercase_auth_method = "voice",           "voice",
   lowercase_auth_method = null,              null,    to_string(lowercase_auth_method))
```

Supported values (per applicable event type):



- **hardware\_token**: Physical device-based authentication, such as RSA token or YubiKey.
- **password** : Standard password-based login.
- **application**: Action approved via an authenticator app, such as push notification.
- **email**: Verification through email link or one-time code.
- **sms**: SMS-based one-time password (OTP) or verification.
- **voice**: Verification via voice call.
- **trusted\_login**: Login from a known or previously trusted device.
- **Generic sso**: Federated authentication using a standard single sign-on provider.
- **null**: No sub-type specified or not applicable.

#### 19. xdm.source.user.identifier

**Authentication Story Field:** `auth_identity_id`

**Type:** string

**Requirement:** Optional

**Description:** This field should contain a unique and consistent identifier for the user associated with the event.

**Example values:**

- `a1b2c3d4-e5f6-7890-ab12-3456789cdedef0`: Directory object GUID, such as Azure AD object ID.
- `S-1-5-21-3623811015-3361044348-30300820-1013`: Windows Security Identifier (SID).

**Best Practice:** Populate with the most persistent and canonical user identifier available from the identity source.

#### 20. xdm.source.user.username

**Authentication Story Field:** `auth_identity_display_name`

**Type:** string

**Requirement:** Optional

**Description:** User Display Name Field. This field should contain the user's name in a human-readable format, typically including the first and last name, such as John Smith.

#### 21. xdm.source.user.user.type

**Authentication Story Field:** `auth_normalized_user.identity_type`

**Type:** string

**Requirement:** Optional

**Description:** Indicates the type of identity associated with the authentication event.

**Supported values (Constants):**

- `XDM_CONST.USER_TYPE_REGULAR ("USER")`
- `XDM_CONST.USER_TYPE_SERVICE_ACCOUNT ("SERVICE ACCOUNT")`
- `XDM_CONST.USER_TYPE_MACHINE_ACCOUNT ("MACHINE ACCOUNT")`

**Example mapping logic:** The following pseudocode is an example only, which must be modified to implement the logic in the correct syntax.

```
if(actor_type in("User"), XDM_CONST.USER_TYPE_REGULAR, actor_type
in("SystemPrincipal"),
XDM_CONST.USER_TYPE_SERVICE_ACCOUNT,actor_type in("IP address"),
XDM_CONST.USER_TYPE_MACHINE_ACCOUNT, to_string(actor_type))
```

#### NOTE:

For more information, see `XDM_CONST.USER_TYPE`.

#### 22. xdm.auth.privilege.level

**Authentication Story Field:** `auth_normalized_user.privilege_level`



Type: string

Requirement: Optional

Description: Represents the privilege level or role of the user during the authentication event, such as admin, user, or guest. Used to assess risk and impact. Should reflect a canonical privilege level.

Supported values (Constants):

- XDM\_CONST.PRIVILEGE\_LEVEL\_GUEST
- XDM\_CONST.PRIVILEGE\_LEVEL\_USER
- XDM\_CONST.PRIVILEGE\_LEVEL\_ADMIN
- XDM\_CONST.PRIVILEGE\_LEVEL\_SYSTEM

Example mapping logic: The following pseudocode is an example only, which must be modified to implement the logic in the correct syntax.

```
if(lowercase_user_type = "user", XDM_CONST.PRIVILEGE_LEVEL_USER,
lowercase_user_type = "guest", XDM_CONST.PRIVILEGE_LEVEL_GUEST,
lowercase_user_type = "admin", XDM_CONST.PRIVILEGE_LEVEL_ADMIN,
lowercase_user_type = "system", XDM_CONST.PRIVILEGE_LEVEL_SYSTEM,
lowercase_user_type = null, null, to_string(lowercase_user_type))
```

#### NOTE:

For more information, see [XDM\\_CONST.PRIVILEGE\\_LEVEL](#).

### 23. xdm.target.resource.id

Authentication Story Field: auth\_target\_id

Type: string

Requirement: Optional

Description: Represents the unique identifier of the accessed resource or application during the authentication event, such as application ID or internal resource ID.

### 24. xdm.target.resource.name

Authentication Story Field: auth\_target

Type: string

Requirement: Optional

Description: Provides the readable name of the resource or application accessed during the event. This should reflect the logical service or platform the user interacted with.

Example values: `Exchange`, `SharePoint`, `ServiceNow`, `HR Portal`, `Okta Admin Portal`

### 25. xdm.source.user.agent

Authentication Story Field: action\_user\_agent

Type: string

Requirement: Optional

Description: Captures the full user-agent string from the client initiating the authentication request. Typically includes information about the browser, operating system, device type, and version details.

Example values: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.86 Safari/537.36

### 26. xdm.network.session\_id

Authentication Story Field: sso\_debug\_data.session\_id

Type: string

Requirement: Optional

Description: Represents the session identifier associated with the user interaction. This value is typically issued by the application or identity platform and is used to group multiple related actions or requests made by the same user or device within a defined time window, commonly referred to as a session.

These can include login, token refresh, or other events that occur as part of a continuous authenticated interaction.



**Example use cases:**

- Tracking session duration
- Linking session start and end events
- Grouping multiple MFA prompts within a single login session

**27. xdm.source.location.city**

**Authentication Story Field:** `action_location.city`

**Type:** string

**Requirement:** Optional

**Description:** Represents the city associated with the source IP address.

**28. xdm.source.location.country**

**Authentication Story Field:** `action_location.country`

**Type:** string

**Requirement:** Optional

**Description:** Identifies the country where the authentication request originated, based on the source IP address.

**Example values:** US, IL, DE, FRANCE

**29. xdm.source.location.region**

**Authentication Story Field:** `action_location.region`

**Type:** string

**Requirement:** Optional

**Description:** Captures the region, province, or state associated with the source IP address.

**Example values:** California, Bavaria, Tel Aviv

**30. xdm.source.location.latitude**

**Authentication Story Field:** `action_location.latitude`

**Type:** float

**Requirement:** Optional

**Description:** Represents the latitude coordinate of the source IP address's estimated physical location.

**Example values:** 40.7128, 48.8566

**31. xdm.source.location.longitude**

**Authentication Story Field:** `action_location.longitude`

**Type:** float

**Requirement:** Optional

**Description:** Represents the longitude coordinate of the source IP address's estimated physical location.

**32. xdm.source.location.continent**

**Authentication Story Field:** `action_location.continent`

**Type:** string

**Requirement:** Optional

**Description:** Indicates the continent associated with the source IP's location, such as Europe, Asia, or North America.

**33. xdm.source.location.timezone**

**Authentication Story Field:** `action_location.timezone`



Type: string

Requirement: Optional

Description: This enables time-based correlation between user activity and local context.

Example values: "Asia/Jerusalem", "America/New\_York", "UTC"

#### 34. xdm.source.host.device.category

Authentication Story Field: auth\_client\_type

Type: string

Requirement: Optional

Description: Represents the operating system family of the device or client that initiated the event.

Supported values:

- Computer : A desktop or laptop device.
- Mobile : A mobile phone or smartphone.
- IoT : An Internet of Things device, such as smart TV or smart speaker.
- Tablet : A tablet computing device.

#### 35. xdm.source.application.name

Authentication Story Field: agent\_extra\_data.browser

Type: string

Requirement: Optional

Description: Represents browser vendor used during the authentication event.

Example values: Chrome , Firefox , Safari , Edge

#### 36. xdm.source.application.version

Authentication Story Field: agent\_extra\_data.browser\_version

Type: string

Requirement: Optional

Description: Represents the version of the browser used during the authentication event.

Example values: Chrome 113 , Firefox 102 , Safari 16.3

#### 37. xdm.source.host.os.family

Authentication Story Field: agent\_os\_type

Type: string

Requirement: Optional

Description: Provides a normalized, high-level abstraction of the operating system associated with the source host. This field enables consistent behavior across different data sources.

Required action: You must explicitly define a mapping logic between the raw field in the original data source that contains the OS information, such as device.os\_name, and normalize into the XDM field.

Common values (Constants):

- XDM\_CONST.OS\_FAMILY\_WINDOWS
- XDM\_CONST.OS\_FAMILY\_MACOS
- XDM\_CONST.OS\_FAMILY\_LINUX
- XDM\_CONST.OS\_FAMILY\_ANDROID
- XDM\_CONST.OS\_FAMILY\_IOS



**NOTE:**

For more information, see [XDM\\_CONST.OS\\_FAMILY](#).

Example mapping logic: The following pseudocode is an example only, which must be modified to implement the logic in the correct syntax.

```
if raw_event.host_os contains "windows"      à XDM_CONST.OS_FAMILY_WINDOWS
if raw_event.host_os contains "mac"          à XDM_CONST.OS_FAMILY_MACOS
if raw_event.host_os contains "linux"        à XDM_CONST.OS_FAMILY_LINUX
if raw_event.host_os contains "android"      à XDM_CONST.OS_FAMILY_ANDROID
if raw_event.host_os contains "ios"          à XDM_CONST.OS_FAMILY_IOS
```

Data Model Rule example logic:

```
if(os contains "windows", XDM_CONST.OS_FAMILY_WINDOWS, os contains "mac", XDM_CONST.OS_FAMILY_MACOS, ...)
```

### 38. xdm.source.host.os

Authentication Story Field: `agent_os_sub_type`

Type: string

Requirement: Optional

Description: This field captures the raw operating system information reported by the source host's telemetry.

Example values: "Microsoft Windows NT 10.0", "Microsoft Windows NT 6.1 (Windows 7)", "Darwin Kernel Version 20.6.0"

### 39. xdm.session.context.id

Authentication Story Field: `auth_correlation_id`

Type: string

Requirement: Optional

Description: Represents a correlation ID tied to a single authentication or access-related request. This ID is generated by the identity provider, such as Azure AD or Okta, and used to link events belonging to a single logical transaction, such as an SSO login flow or token issuance.

Example values: "25423545-6364-5423-3232-42343760"

**NOTE:**

While `xdm.network.session_id` aggregates multiple user actions within a broader session window, `xdm.session.context.id` is used to correlate events that belong to a single authentication request or transaction.

### 40. xdm.time

Authentication Story Field: `generatedTime`

Type: timestamp

Description: Represents the timestamp of when the event occurred. This field is essential for event sequencing and correlation. This field is automatically mapped.

Example values:

- "2025-05-26T14:45:32Z" (UTC format ISO 8601)
- "2025-05-26T14:45:32.123Z" (UTC with millisecond precision),
- "2025-05-26 14:45:32" (Standard datetime format non-ISO)

## 8.3 | Fair Usage policy for Cortex AgentiX

To ensure the reliability, efficiency, and availability of Cortex AgentiX for all the users, we expect our customers to use it fairly, reasonably, and in a manner that does not adversely impact the product's overall system performance. Cortex AgentiX offers various features for connecting external resources to Cortex AgentiX, including without limitation, frequency and/or volume of data ingestion, number of connected data sources, and API usages. Overuse or misuse of these features may adversely affect the reliability, efficiency, and/or availability of Cortex AgentiX.

You are therefore required to utilize a reasonable volume of data ingestion, number of connected data sources, and API usage, based on your number of cloud assets protected by Cortex AgentiX ("Fair Usage Policy"). If, in our sole and reasonable discretion, we determine that your usage of Cortex AgentiX violates this Fair Usage Policy, we reserve the right to take appropriate action regarding such use, including without limitation, limiting the frequency and/or volume of data ingestions, limiting the number of connected data sources, and/or limiting the API usage, to bring your usage of Cortex AgentiX in alignment with this Fair Usage policy.

