cisco Live!

Let’s go

The bridge to possible

Securing APIs from Left to Right

And Everywhere in Between

Shannon McFarland, VP, Engineering, Cisco DevNet
CCIE#5245
@eyepv6

DEVNET-2703

Agenda

- API Level Set
- Open Worldwide Application Security Project (OWASP) Top 10 API Security Risks
- What is the API Pipeline?
- Prevent Common API Security Issues Across the Pipeline

Just published!: https://blogs.cisco.com/developer/securing-apis-from-left-to-right-and-everywhere-in-between

Other Sessions

- DEVNET-2220: Harnessing the Power of APIs in Artificial Intelligence
- DEVNET-2626: API-Lifecycle Pipelines Uncovered: Using automation for quality assurance
- DEVLIT-2209: Nobody puts docs in a corner
- DEVNET-2710: API design principals and considerations for massive scale
- DEVWKS-2285: Introduction to APIClarity – A Wireshark for APIs
- DEVWKS-2706: Orchestrating traffic navigation and API insights in Cloud Native Apps
- DEVNET-2040: Use Case Centric APIs
- DEVNET-2011: Cloud Native Application Observability API

API Level Set

What is an API?

- Application Programming Interface (API) - A set of rules, protocols, and tools that allow different software applications to communicate and interact with each other

- APIs enable seamless integration of functionalities, data, and services across various platforms, systems, and devices

- Key Components:
  - **Endpoints:** Specific URLs or routes that serve as access points for requesting data or actions from the API
  - **Requests:** Client applications send requests to the API to retrieve or manipulate data
  - **Responses:** The API sends back responses containing the requested data or confirmation of an action
  - **Methods:** Actions that can be performed through the API, such as GET (retrieve data), POST (create data), PUT (update data), and DELETE (remove data)

© Tierney / stock.adobe.com

© Mongta Studio / stock.adobe.com

Developers Create APIs – We Use Them

API Security Challenges

Consistency

Scale

Visibility

AuthN/AuthZ

Granularity

Operations

OWASP Top 10

API Security Risks

OWASP Top 10 API Security Risks

API1:2023 Broken Object Level Authorization (BOLA)

- Allowing unauthorized access to objects/data

API2:2023 Broken Authentication

- Allowing attackers access to an application (and ultimately, data)

API3:2023 Broken Object Property Level Authorization

- Similar to BOLA, allowing unauthorized access to data

API4:2023 Unrestricted Resource Consumption

- Allowing unrestricted access to application resources (e.g. DoS attacks)
- Can be very costly (e.g. paid-tier cloud resource consumption)

https://owasp.org/API-Security/editions/2023/en/0x11-t10/

OWASP Top 10 API Security Risks

API5:2023 Broken Function Level Authorization (BFLA)

- Allowing unauthorized access to application functionality (and ultimately, data)

API6:2023 Unrestricted Access to Sensitive Business Flows

- Vulnerability to automated abuse of application transactions (e.g. ticket sales, thread comments)
- “Bad bots” – can be used for DoS attacks or to try and circumvent security or gain access to high-demand items

API7:2023 Server Side Request Forgery (SSRF)

- Server brokers a request to a remote resource via unvalidated URI from the user (“request spoofing)

OWASP Top 10 API Security Risks

API8:2023 Security Misconfiguration

- Any weak or misconfigured security in an application opens attack surfaces

API9:2023 Improper Inventory Management

- Undocumented APIs open attack surfaces

API10:2023 Unsafe Consumption of APIs

- Weak security in 3rd party APIs can allow access to data

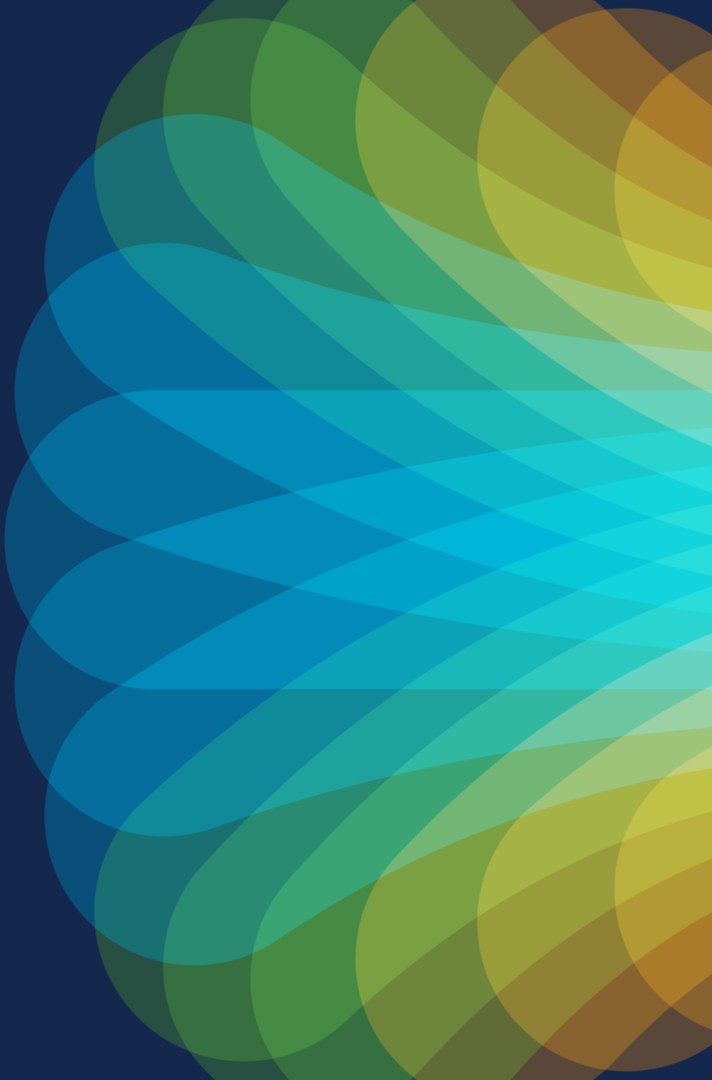
OWASP Top 10 Main Targets

© ArtemisDiana / Shutterstock.com

- Sensitive data breaches
  - Personally Identifiable Information (PII) like name, address, username, passwords, SSN, financial data
  - Could be used for identity theft, to access accounts or sold on dark web
- Access to high-demand items for resale (scalping)
- Application instability/unavailability
- Extortion, ransom demands, espionage, voting interference, political instability

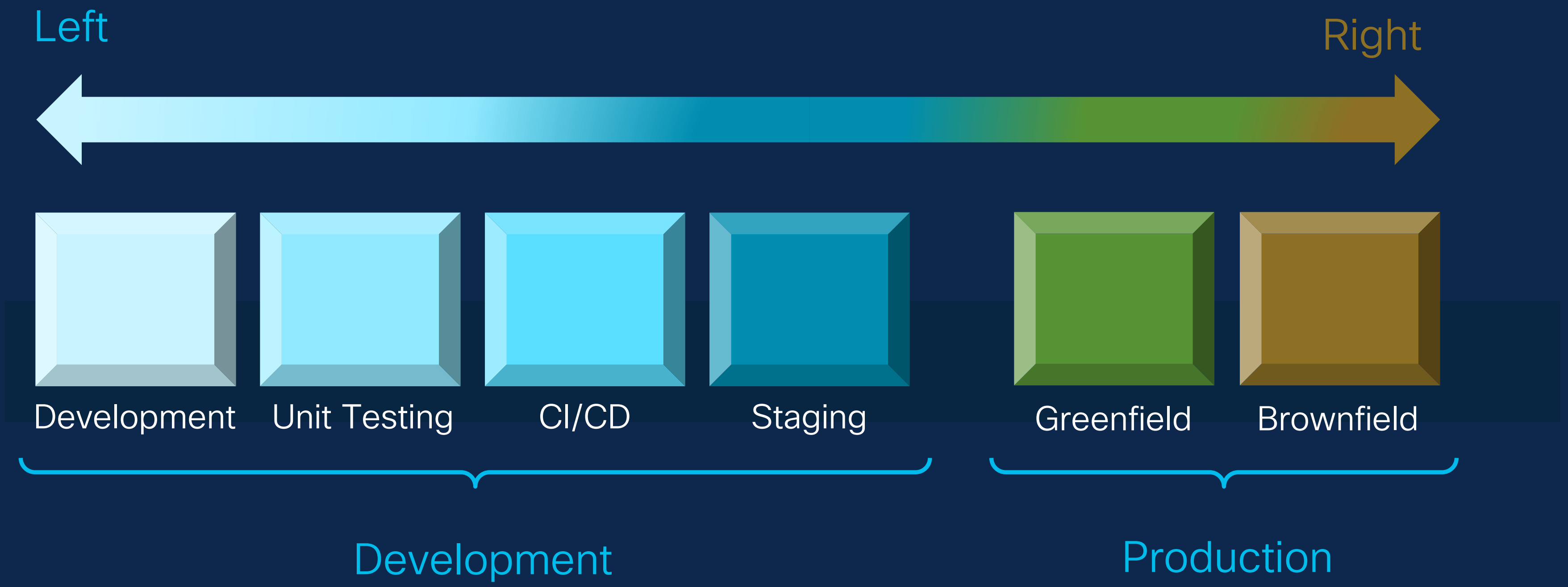
AI will make it increasingly easier/faster to circumvent API security, but AI will also make it easier to detect and remediate attack paths

What is the API Pipeline?

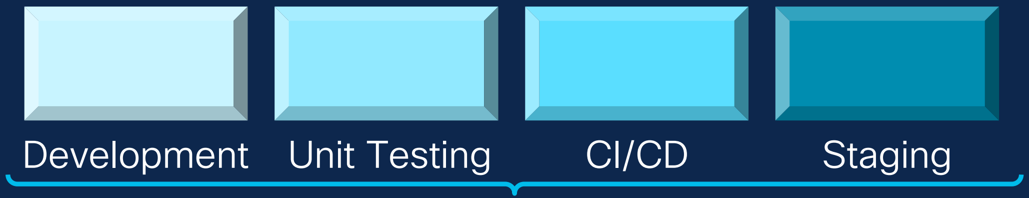


API Pipeline

Spans the entire API lifecycle

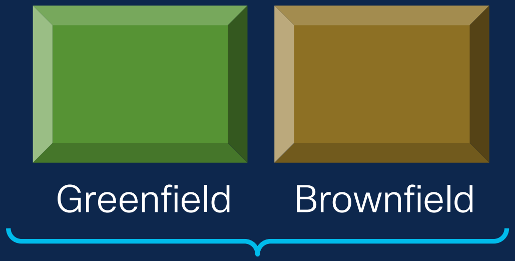


Development



- Development is where APIs are created and hardened against potential threats in simulated testing environments
- The ability to tighten API security starts with design decisions in development
- API development includes:
  - Using IDEs/AI copilots to write the client/server code
  - Writing an OpenAPI spec
  - Configuring API security for endpoints
  - Deciding which 3rd party APIs and applications to interact with
  - Unit testing (aka mock testing)
  - CI/CD testing and benchmarks
  - Deploying and testing in a staging environment (like production, but in-house, isolated)

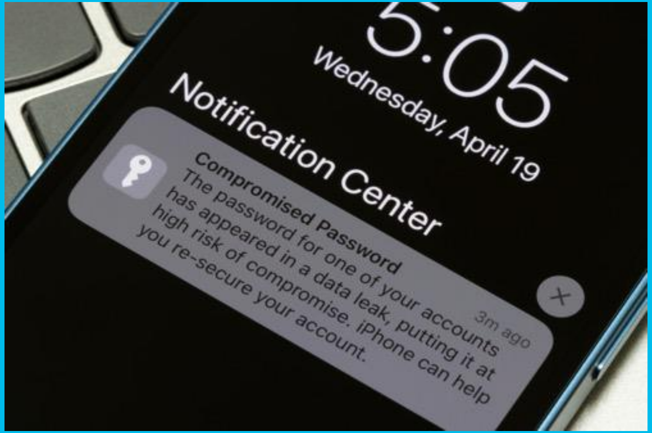
Production



- Production is where your APIs become exposed to security threats 24/7
- Security problems found in production go back through the stages of development, take time to mitigate
- A first-time deployment, with no prior history or versions in existence, is called a greenfield installation
- Upgrades to a deployed application are called brownfield installations and need to account for API versioning, backwards compatibility, etc.
  - Canary deployments are a subset of brownfield – API traffic is split between the existing application and a new version

Prevent Common API Security Issues Across the Pipeline

Broken Object Level Authorization (BOLA)



Compromised Password
The password for one of your accounts has appeared in a data leak, putting it at high risk of compromise. iPhone can help you re-secure your account.

© Tada Images / Shutterstock.com

- BOLA happens when data is accessed by end users without proper authorization
- OWASP issues API1:2023 and API3:2023 are about BOLA
- Since data (especially PII) is a major target of breaches, any unauthorized access is a huge security problem

BOLA

How To Prevent?

- Develop a solid authorization model in your application, and between microservices
  - Never expose objects/data in a response without authorization
  - Use Random UUIDs
  - Zero-Trust
  - Fine-Grained authorization – Policy, attributes, etc.
- Run OpenAPI spec (OAS) linters in development and CI/CD
  - Examples: Spectral (including OWASP Top 10 ruleset), Postman, Speccy, Panoptica
  - Many linters have IDE integrations (e.g. VSCode)
- Run mock API traffic in unit testing and CI/CD, try to access data without authorization
  - Examples: WireMock, Microcks, Postman, RestAssured, SoapUI
- Panoptica can run a fuzzer against API endpoints in CI/CD and staging, sends bad input into APIs, flags any unexpected access to data
- Run Dynamic Application Security Testing (DAST) tools in staging and production
  - Panoptica can detect BOLA issues with real-time API traffic

Broken Function Level Authorization (BFLA)

© jijomathaidesigners / Shutterstock.com

- Happens when application functionality is accessed by end users without proper authorization (BOLA is about accessing data, BFLA is about accessing functionality)
- BFLA is OWASP issue API5:2023
- Gaining unauthorized access to any application functionality can ultimately lead to data breaches
- BFLA includes authorization between microservices in your application

BFLA

How To Prevent?

- Develop a solid authorization model in your application, and between microservices
  - Zero-Trust
  - Fine-Grained authorization – Policy, attributes, etc.
    - Context-aware of API methods: POST allowed, but not DELETE
- Run mock API traffic in unit testing and CI/CD, try to access functionality between microservices without authorization
- Run DAST tools in staging and production
    - Panoptica can learn the authorization model, detect BFLA issues

Weak Authentication

© KT Stock photos / Shutterstock.com

- Weak authentication is easier to compromise
- Examples
  - Weak passwords
    - Easily guessed
    - Reused across accounts
  - Weak endpoint security
    - Using HTTP, not HTTPs
  - Weak encryption
- Weak auth is OWASP issue API2:2023
- Could give threat actors access to accounts, data

Weak Authentication

How To Prevent?

- Develop secure endpoints with encryption enabled (e.g. HTTPs)
- Require strong passwords and MFA
- Run OAS linters in development and CI/CD
  - Spectral with OWASP Top 10 ruleset can flag insecure endpoints
- Run mock API traffic in unit testing and CI/CD using weak authentication, try to gain access
- Run DAST tools in staging and production
  - Panoptica can detect weak authentication issues

Shadow APIs

© mkfilm / Shutterstock.com

- Shadow APIs are unknown or undocumented APIs (not included in an OpenAPI spec)
- OWASP issue API9:2023
- Undocumented APIs in your application are a security risk you don't know you have
- As your application evolves, shadow API security will likely not keep up
- Some shadow APIs can be forgotten entirely, exposing an ongoing security loophole or backdoor into your application

Shadow APIs

How To Prevent?

- Inventory and Document all APIs for your application (Example: OpenAPI spec)
- Run DAST tools in staging and production
  - Panoptica can flag shadow APIs in staging and production, and reconstruct OpenAPI specs to document them properly

Zombie APIs

© FOTOKITA / Shutterstock.com

- Zombie APIs are deprecated APIs that are still active
- OWASP issue API9:2023
- They occur in brownfield and canary production environments, where multiple API versions are in use
- They are a problem because they will likely not evolve with your application and could receive less scrutiny from a security standpoint
- As with shadow APIs, zombies could provide a backdoor into your application

Zombie APIs

How To Prevent?

- Any API in the OpenAPI spec that is labelled “deprecated” is a zombie
- Remove support for zombie APIs as soon as possible
- Panoptica can flag zombie API calls in staging and production

Weak 3rd Party Authentication

© ZinetroN / Shutterstock.com

- 3rd party APIs could allow access to application data (e.g. databases, S3 buckets, etc.)
- Weak 3rd party API authentication makes data vulnerable (even if application APIs are secure)
- OWASP issue API10:2023 addresses weak 3rd party API security

Weak 3rd Party Authentication

How To Prevent?

- Keep an inventory of all 3rd party APIs called during development
- Make sure 3rd party APIs are secure
- Panoptica can give a security score to 3rd party APIs
- Use cloud security scanners in staging and production to detect weak auth
  - AWS Config, Azure Automation & Control, GCP Cloud Asset Inventory, Cloudquery

Data Injection

© Mega Pixel / Shutterstock.com

- Data injection can allow threat actors to pass malicious data, configurations or programs via an API into applications
- Malicious data injection could allow access to data (e.g. BOLA) or make an application unstable

Data Injection

How To Prevent?

- During development, include strict type checking <‘str’ not ‘float’> and input validation in API processing
- Establish upper limits on size and quantity of data that can be input
- Run OpenAPI linters in development and CI/CD
- Run mock API traffic in unit testing and CI/CD, try to inject invalid data
- Panoptica can run a fuzzer against API endpoints in CI/CD and staging, which attempts to send bad data into APIs
- Panoptica can dynamically compare API traffic against the OAS and flag data discrepancies (including spec drift) in staging and production

Code Injection

© solarseven / Shutterstock.com

- Code injection is where undesirable code is added to an application
- IDE plugins and AI co-pilots are increasingly used to generate API client/server code
- These plugins and co-pilots could inject “bad” code into your application, which can have unintended or even malicious side-effects
  - For example, a rogue API could be injected into your application creating backdoor access.

Code Injection

How To Prevent?

- Any generated code must be verified during development
- Thorough/expert code reviews will help, but can’t catch everything
- Images scans can search for any Common Vulnerabilities and Exposures (CVEs) in CI/CD, staging and production environments
  - Panoptica scans both Kubernetes container and virtual machine images
- In staging and production, run dynamic API security tools to scan for any rogue APIs - Panoptica has this capability

Panoptica—Comprehensive Code to Cloud Security – www.panoptica.app

Code + Build Security

Prioritize and Remediate:

- Full Development Lifecycle
- Governance Policies
- Infrastructure as Code

Cloud Security Posture Management

Automate and Simplify:

- Compliance Monitoring
- Resource Management

Cloud Workload Protection

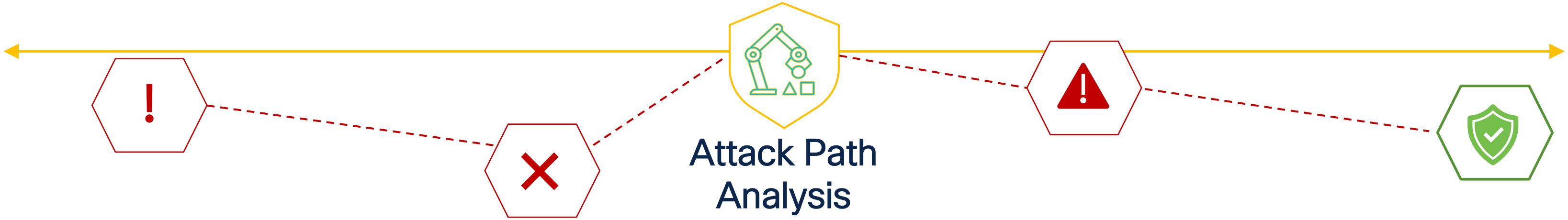
Continuous Risk Management:

- Virtual Machines
- Containers
- Serverless

API Security

Assess and Monitor:

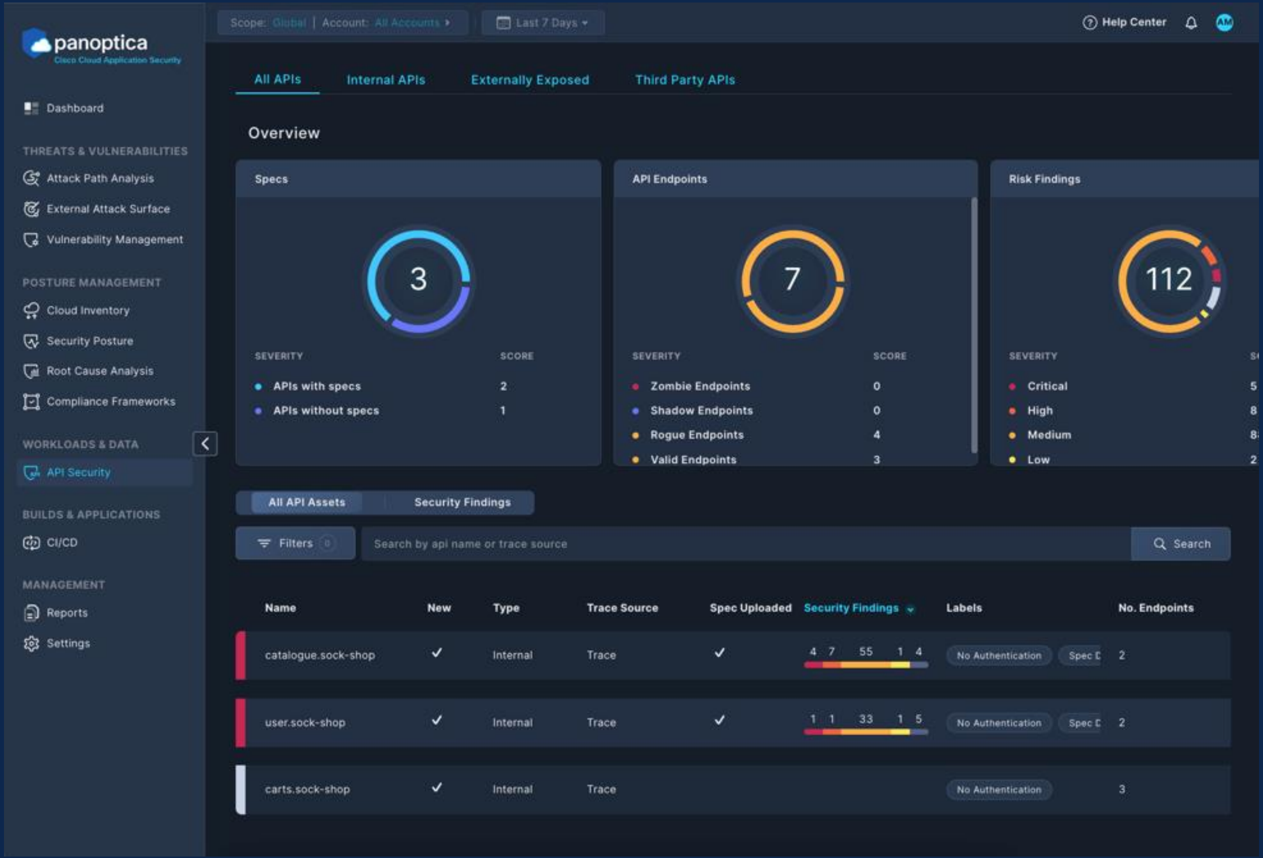
- Internal & External APIs
- API Tokens



Attack Path Analysis

Prioritize with precision.
Remediate the risks that matter—first.

Panoptica API Security UI



Scope: Global | Account: All Accounts

Last 7 Days

Overview

Specs 3

SEVERITY	SCORE
● APIs with specs	2
● APIs without specs	1

API Endpoints 7

SEVERITY	SCORE
● Zombie Endpoints	0
● Shadow Endpoints	0
● Rogue Endpoints	4
● Valid Endpoints	3

Risk Findings 112

SEVERITY	S
● Critical	5
● High	8
● Medium	8
● Low	2

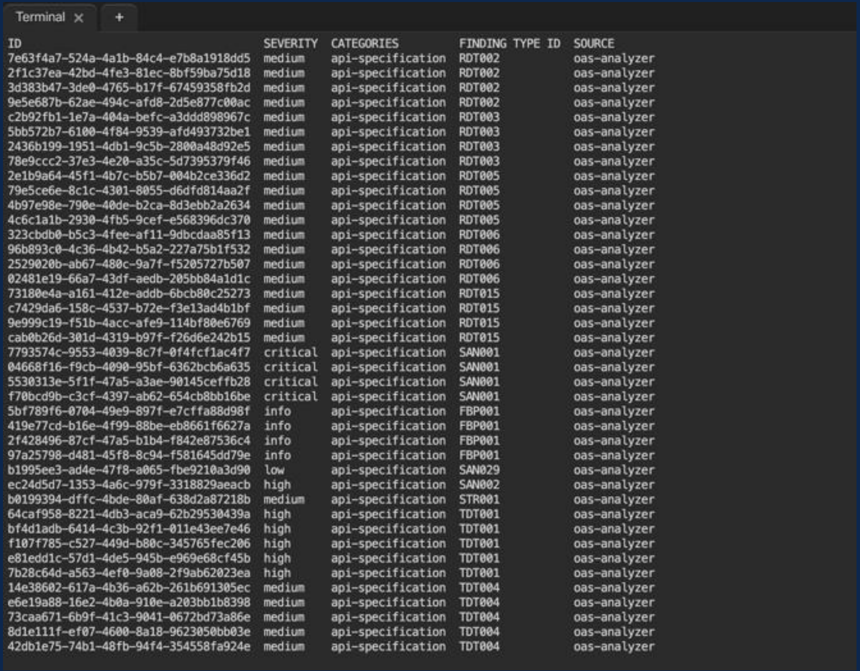
Search by api name or trace source

Name	New	Type	Trace Source	Spec Uploaded	Security Findings	Labels	No. Endpoints
catalogue.sock-shop	✓	Internal	Trace	✓	4 7 55 1 4	No Authentication Spec D	2
user.sock-shop	✓	Internal	Trace	✓	1 1 33 1 5	No Authentication Spec D	2
carts.sock-shop	✓	Internal	Trace			No Authentication	3

Panoptica API Security CLI

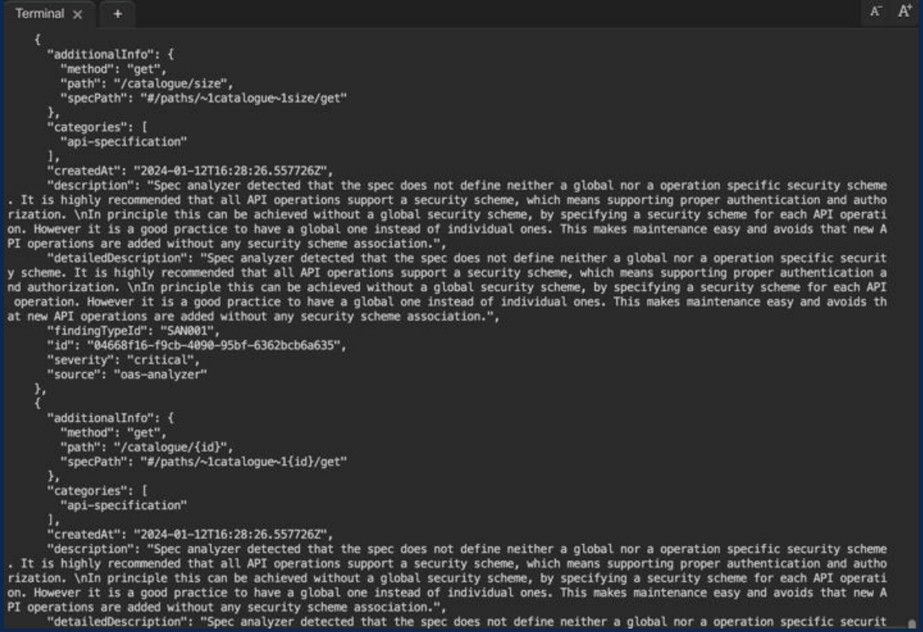
With the CLI, you can:

- Run OpenAPI specification analysis
- Get security scores for 3rd party APIs
- Run a fuzzing job



```
ID                                    SEVERITY  CATEGORIES         FINDING TYPE ID  SOURCE
7e63f4a7-524a-4a1b-84c4-e7b8a1918dd5  medium    api-specification  RDT002           oas-analyzer
2f1c37ea-42bd-4fe3-81ec-8bf59ba75d18  medium    api-specification  RDT002           oas-analyzer
3d383b47-3de0-4765-b17f-67459358fb2d  medium    api-specification  RDT002           oas-analyzer
9e5e687b-62ae-494c-afd8-2d5e877c00ac  medium    api-specification  RDT002           oas-analyzer
c2b92fb1-1e7a-404a-befc-a3ddd898967c  medium    api-specification  RDT003           oas-analyzer
5bb572b7-6100-4f84-9539-afd493732be1  medium    api-specification  RDT003           oas-analyzer
2436b199-1951-4db1-9c5b-2800a48d92e5  medium    api-specification  RDT003           oas-analyzer
78e9ccc2-37e3-4e20-a35c-5d7395379f46  medium    api-specification  RDT003           oas-analyzer
2e1b9a64-45f1-4b7c-b5b7-004b2ce336d2  medium    api-specification  RDT005           oas-analyzer
79e5ce6e-8c1c-4301-8055-d6dfd814aa2f  medium    api-specification  RDT005           oas-analyzer
4b97e98e-790e-40de-b2ca-8d3ebb2a2634  medium    api-specification  RDT005           oas-analyzer
4c6c1a1b-2930-4fb5-9cef-e568396dc370  medium    api-specification  RDT005           oas-analyzer
323cbdb0-b5c3-4fee-af11-9dbcdaa85f13  medium    api-specification  RDT006           oas-analyzer
96b893c0-4c36-4b42-b5a2-227a75b1f532  medium    api-specification  RDT006           oas-analyzer
2529020b-ab67-480c-9a7f-f5205727b507  medium    api-specification  RDT006           oas-analyzer
02481e19-66a7-43df-aedb-205bb84a1d1c  medium    api-specification  RDT006           oas-analyzer
73180e4a-a161-412e-addb-6bcb80c25273  medium    api-specification  RDT015           oas-analyzer
c7429da6-158c-4537-b72e-f3e13ad4b1bf  medium    api-specification  RDT015           oas-analyzer
9e999c19-f51b-4acc-afe9-114bf80e6769  medium    api-specification  RDT015           oas-analyzer
cab0b26d-301d-4319-b97f-f26d6e242b15  medium    api-specification  RDT015           oas-analyzer
7793574c-9553-4039-8c7f-0f4fcf1ac4f7  critical  api-specification  SAN001           oas-analyzer
04668f16-f9cb-4090-95bf-6362bcb6a635  critical  api-specification  SAN001           oas-analyzer
5530313e-5f1f-47a5-a3ae-90145ceffb28  critical  api-specification  SAN001           oas-analyzer
f70bcd9b-c3cf-4397-ab62-654cb8bb16be  critical  api-specification  SAN001           oas-analyzer
5bf789f6-0704-49e9-897f-e7cffa88d98f  info      api-specification  FBP001           oas-analyzer
419e77cd-b16e-4f99-88be-eb8661f6627a  info      api-specification  FBP001           oas-analyzer
2f428496-87cf-47a5-b1b4-f842e87536c4  info      api-specification  FBP001           oas-analyzer
97a25798-d481-45f8-8c94-f581645dd79e  info      api-specification  FBP001           oas-analyzer
b1995ee3-ad4e-47f8-a065-fbe9210a3d90  low       api-specification  SAN029           oas-analyzer
ec24d5d7-1353-4a6c-979f-3318829aeacb  high      api-specification  SAN002           oas-analyzer
b0199394-dffc-4bde-80af-638d2a87218b  medium    api-specification  STR001           oas-analyzer
64caf958-8221-4db3-aca9-62b29530439a  high      api-specification  TDT001           oas-analyzer
bf4d1adb-6414-4c3b-92f1-011e43ee7e46  high      api-specification  TDT001           oas-analyzer
f107f785-c527-449d-b80c-345765fec206  high      api-specification  TDT001           oas-analyzer
e81edd1c-57d1-4de5-945b-e969e68cf45b  high      api-specification  TDT001           oas-analyzer
7b28c64d-a563-4ef0-9a08-2f9ab62023ea  high      api-specification  TDT001           oas-analyzer
14e38602-617a-4b36-a62b-261b691305ec  medium    api-specification  TDT004           oas-analyzer
e6e19a88-16e2-4b0a-910e-a203bb1b8398  medium    api-specification  TDT004           oas-analyzer
73caa671-6b9f-41c3-9041-0672bd73a86e  medium    api-specification  TDT004           oas-analyzer
8d1e111f-ef07-4600-8a18-9623050bb03e  medium    api-specification  TDT004           oas-analyzer
42db1e75-74b1-48fb-94f4-354558fa924e  medium    api-specification  TDT004           oas-analyzer
```

Documentation: https://docs.panoptica.app/docs/api-security-cli-jobs



```
    {
      "additionalInfo": {
        "method": "get",
        "path": "/catalogue/size",
        "specPath": "#/paths/~1catalogue~1size/get"
      },
      "categories": [
        "api-specification"
      ],
      "createdAt": "2024-01-12T16:28:26.557726Z",
      "description": "Spec analyzer detected that the spec does not define neither a global nor a operation specific security scheme
. It is highly recommended that all API operations support a security scheme, which means supporting proper authentication and autho
rization. \nIn principle this can be achieved without a global security scheme, by specifying a security scheme for each API operati
on. However it is a good practice to have a global one instead of individual ones. This makes maintenance easy and avoids that new A
PI operations are added without any security scheme association.",
      "detailedDescription": "Spec analyzer detected that the spec does not define neither a global nor a operation specific securit
y scheme. It is highly recommended that all API operations support a security scheme, which means supporting proper authentication a
nd authorization. \nIn principle this can be achieved without a global security scheme, by specifying a security scheme for each API
 operation. However it is a good practice to have a global one instead of individual ones. This makes maintenance easy and avoids th
at new API operations are added without any security scheme association.",
      "findingTypeId": "SAN001",
      "id": "04668f16-f9cb-4090-95bf-6362bcb6a635",
      "severity": "critical",
      "source": "oas-analyzer"
    },
    {
      "additionalInfo": {
        "method": "get",
        "path": "/catalogue/{id}",
        "specPath": "#/paths/~1catalogue~1{id}/get"
      },
      "categories": [
        "api-specification"
      ],
      "createdAt": "2024-01-12T16:28:26.557726Z",
      "description": "Spec analyzer detected that the spec does not define neither a global nor a operation specific security scheme
. It is highly recommended that all API operations support a security scheme, which means supporting proper authentication and autho
rization. \nIn principle this can be achieved without a global security scheme, by specifying a security scheme for each API operati
on. However it is a good practice to have a global one instead of individual ones. This makes maintenance easy and avoids that new A
PI operations are added without any security scheme association.",
      "detailedDescription": "Spec analyzer detected that the spec does not define neither a global nor a operation specific securit
```

Summary

- API security is not a once-and-done effort
- API security must be top-of-mind from design-to-code-to-deployment
- The OWASP Top 10 list is a great starting point, but there are other threats such as data and code injection dangers
- There are multiple tools and solutions available to help secure APIs during each stage of the API Pipeline
- Check out Cisco’s Cloud Application Security Product – Panoptica: https://www.panoptica.app/

Other Sessions

- DEVNET-2220: Harnessing the Power of APIs in Artificial Intelligence
- DEVNET-2626: API-Lifecycle Pipelines Uncovered: Using automation for quality assurance
- DEVLIT-2209: Nobody puts docs in a corner
- DEVNET-2710: API design principals and considerations for massive scale
- DEVWKS-2285: Introduction to APIClarity – A Wireshark for APIs
- DEVWKS-2706: Orchestrating traffic navigation and API insights in Cloud Native Apps
- DEVNET-2040: Use Case Centric APIs
- DEVNET-2011: Cloud Native Application Observability API

Questions?

CISCO DevNet

WE ARE CISCO DEVNET 10

Continue to Learn, Code and Build with Cisco DevNet!

Get access to an exclusive learning module filled with digital learning opportunities on topics including Security and more.

Scan QR Code to get started.

The bridge to possible

Thank you

cisco Live!

Let’s go