

**Microsoft Defender ATP** Connector in Azure Sentinel

# Microsoft Defender ATP Connector

Content

1. What is Microsoft Defender ATP?
2. Describe the onboarding process.
3. What are Microsoft Defender ATP Alerts?
4. Sentinel onboarding process.
5. Plans for private preview.
6. Plans for GA.

# What is Microsoft Defender ATP?

Microsoft Defender Advanced Threat Protection is a platform designed to help enterprise networks prevent, detect, investigate, and respond to advanced threats.

Microsoft Defender ATP uses the following combination of technology built into Windows 10 and Microsoft's robust cloud service:

* **Endpoint behavioral sensors**: Embedded in Windows 10, these sensors collect and process behavioral signals from the operating system and sends this sensor data to your private, isolated, cloud instance of Microsoft Defender ATP.
* **Cloud security analytics**: Leveraging big-data, machine-learning, and unique Microsoft optics across the Windows ecosystem, enterprise cloud products (such as Office 365), and online assets, behavioral signals are translated into insights, detections, and recommended responses to advanced threats.
* **Threat intelligence**: Generated by Microsoft hunters, security teams, and augmented by threat intelligence provided by partners, threat intelligence enables Microsoft Defender ATP to identify attacker tools, techniques, and procedures, and generate alerts when these are observed in collected sensor data.

For more information visit: [documentation](https://docs.microsoft.com/en-us/windows/security/threat-protection/microsoft-defender-atp/microsoft-defender-advanced-threat-protection).

## Microsoft Defender ATP Portal

Enterprise security teams can use Microsoft Defender Security Center to monitor and assist in responding to alerts of potential advanced persistent threat (APT) activity or data breaches.

You can use the portal to:

* View, sort, and triage alerts from your endpoints
* Search for more information on observed indicators such as files and IP Addresses
* Change Microsoft Defender ATP settings, including time zone and review licensing information.

There are 2 portals available:

1. PPE - [https://wdatpgw.securitycenter.windows.com](https://wdatpgw.securitycenter.windows.com/)
   * Username - [admin@aad171.ccsctp.net](mailto:admin@aad171.ccsctp.net)
   * Password – [key vault](https://ms.portal.azure.com/?flight=1#@microsoft.onmicrosoft.com/asset/Microsoft_Azure_KeyVault/Secret/https://asi-dev-secrets.vault.azure.net/secrets/DevDataConnectorTestUser/9227994cc11b409ca0ab9ca29ced581a)
2. PRD - <https://securitycenter.windows.com>
   * Username – Admin5@SentinelE5.onmicrosoft.com
   * Password – [key vault](https://ms.portal.azure.com/?flight=1#@microsoft.onmicrosoft.com/resource/subscriptions/78ffdd91-611e-402f-8a7e-7ab0b209b7c6/resourceGroups/Common-EUS/providers/Microsoft.KeyVault/vaults/asi-dev-secrets/secrets) (Secret: AmbaWolvesE5-Admin)

## Onboarding to Microsoft Defender ATP

To use Microsoft Defender ATP, you should buy Microsoft 365 E5 license.  
For more information about the minimum requirements open: [link](https://docs.microsoft.com/en-us/windows/security/threat-protection/microsoft-defender-atp/licensing).

You can ask Microsoft Defender ATP team to send you a promo code (valid for 1 year). They can send both PPE promo code and PRD. For PPE, they will have to manually authorize your tenant during the onboarding process.

After buying the license, you will have to onboard Microsoft Defender ATP portal. To do that, just open it for the first time and you will be asked to perform the onboarding process and to select the tenant region.

Once onboarded, you will be asked to connect your first machine. It is a simple process involves downloading a script to your machine and simulate an alert. Please connect using a virtual machine with Windows 10.

To simulate more alerts, you can use the **script (link to script)**. Please do not run this script on your local machine, or else, someone from India will contact you and start investigating your (true story).

To verify that everything works right. Go to the alert queue and verify that you received the alerts:

1. PPE - <https://wdatpgw.securitycenter.windows.com/alertsQueue>
2. PRD - <https://securitycenter.windows.com/alertsQueue>

### Onboarding using Azure Security Center

It is possible to onboard to Microsoft Defender ATP using Azure Security Center.

Windows Defender ATP in Security Center supports detection on Windows Server 2016, 2012 R2, and 2008 R2 SP1 operating systems in a Standard service subscription.

When you use Azure Security Center to monitor servers, a Windows Defender ATP tenant is automatically created, and the Windows Defender ATP data is stored in Europe by default. If you need to move your data to another location, you need to contact Microsoft Support to reset the tenant.

For more information visit: <https://docs.microsoft.com/en-us/azure/security-center/security-center-wdatp>

## Microsoft Defender ATP Alerts

Microsoft Defender ATP alerts has a different scheme than AlertV3. This scheme can be converted to AlertV3.

Microsoft Defender ATP address an Alert as a collection of events. Those events have the same ‘VendorOriginalId’ which can be used for correlation. Those events might have different: Severity, Description, TimeGenerated and Entities.  
The entities field is an array of entities. Only the relevant entities of the event will be sent. It is important to note that no entities will be removed at any point.

There are several consumers to those alerts:

1. MTP – Microsoft Defender ATP export their alerts to an EventHub in AlertV3 scheme. MTP receive those alerts using Scuba routing rule and address only the ‘new’ alerts. The ‘update’ alerts are being dropped. In MTP portal they present the first alert and redirect the user to Microsoft Defender ATP portal for further investigation. The ProviderName of those alerts is ‘MDATP’.
2. Azure Security Center – Microsoft Defender ATP export their alerts to an EventHub in Microsoft defender ATP scheme. ASC convert this scheme to AlertV3. During this process ASC set the ProviderName of the alert to ‘WDATP’ and add additional fields such as WorkspaceSubscriptionId and WorkspaceResourceGroup.  
   The conversion code can be found at: [link](https://msazure.visualstudio.com/One/_git/SecEng-Scuba?path=%2Fsrc%2FPlatform%2FDetectionsPublisher%2FDetectionsPublisher.Logic%2FWindowsDefenderAtpDetectionToAlertV3Maper%2FWindowsDefenderAtpDetectionToAlertV3Mapper.cs&version=GBmaster&line=312&lineStyle=plain&lineEnd=312&lineStartColumn=31&lineEndColumn=44).  
   In ASC portal, they present Microsoft Defender ATP alerts as a collection of alerts grouped by a CorrelationId. They also present a link to continue investigation in Microsoft Defender ATP portal.

The alerts schemes:

* [Microsoft Defender ATP scheme](https://docs.microsoft.com/en-us/windows/security/threat-protection/microsoft-defender-atp/api-portal-mapping).
* [AlertV3 scheme](https://aka.ms/ascalertcontractsdocumentation).

# Microsoft Defender ATP Azure Sentinel Connector

The connector is available from Azure Sentinel data connectors.

To connect Microsoft Defender ATP, the user must be a Tenant Admin, have Microsoft Defender ATP and onboarded to their portal.

By pressing the ‘connect’ button, an API call to Microsoft Defender ATP enablement API will be sent. After successful enablement, a second API call to Sentinel Connectors RP will be performed to add Microsoft Defender ATP Scuba routing rule.

**(Picture of the portal)**

### About the connector model

* Connectivity – The connectivity of the connector is defined by whether the connector exists in the Connectors RP and whether Microsoft Defender ATP enabled Sentinel Onboarding.
* Pre-Requisites –
  + Read + Write permissions to the workspace.
  + Tenant Admin or Security Admin.
  + License to Microsoft Defender ATP and onboarded to their portal.
* Base query – SecurityAlert | where ProviderName == “MDATP”

For more information about the connector model visit: [Sentinel Repo - MdatpDataModel](https://msazure.visualstudio.com/One/_git/ASI-Portal?path=%2Fsrc%2FASIExtension%2FClient%2FData%2FDataConnectorsV3%2FConnectors%2FMdatpDataModel.ts&version=GBmaster).

### About Microsoft Defender ATP enablement API

* **GET https://threatintel-eus-prd.cloudapp.net/sentinelonboarding/tenanturl**
  + Description: use to return the tenant regional URL as been selected during Microsoft Defender ATP onboarding.
  + Authentication: tenant-token or user-token.
  + Returns: { Url : <TenantRegionalUrl>}
  + Example: {"Url":"https://threatintel-weu.securitycenter.windows.com/"}
* **GET <TenantRegionalUrl>/sentinelonboarding/status**
  + Description: use to return the Sentinel onboarding status of the tenant.
  + Authentication: tenant-token or user-token.
  + Returns: { IsTenantActive: Boolean, IsSentinelEnabled: Boolean }
    - IsTenantActive – indicates whether the tenant has a license and onboarded to Microsoft Defender ATP.
    - IsSentinelEnabled – indicates whether the customer gave consent to export Microsoft Defender ATP alerts from Microsoft Defender ATP to Sentinel.
  + Example: {"IsSentinelEnabled": false,"IsTenantActive": true}
* **POST <TenantRegionalUrl>/sentinelonboarding/enable**
  + Decription: use to enable export of Microsoft Defender ATP alerts to Sentinel.
  + Authentication: user-token.
* **PATCH <TenantRegionalUrl>/sentinelonboarding/disable**
  + Description: use to disable the export of Microsoft Defender ATP alerts to sentinel.
  + Authentication: user-token.

# Microsoft Defender ATP Enablement pipeline (Private Preview)

The private preview implementation is based on MTP. Microsoft Defender ATP implemented a similar API for MTP onboarding. When a user onboard to MTP, they export Microsoft Defender ATP alerts in AlertV3 scheme to an EventHub. This EventHub is being consumed by Scuba which redirects the alerts according to MTP routing rules.

Since Microsoft Defender ATP already implemented export to EventHub for MTP and due to time limitation, we re-used the same pipeline for our needs. Microsoft Defender ATP developed the enablement API for sentinel which enable the alerts flow for Sentinel. MTP onboarding and Sentinel onboarding are two different APIs and they do not affect each other.

To use Microsoft Defender ATP enablement API, we had to gain user consent. User consent should be gain through AAD. It works by adding new permissions to the application and by asking the user to login and accept. This process grants the application the ability to perform the desired API calls.

To achieve a user-token for Microsoft Defender ATP, we had to use on-behalf-of authentication. This process allows our application to request user-token from AAD with a different audience than the original token. This method should be performed using a user-token.

There were 2 options to achieve that token:

1. Through Sentinel RP – Due to limitation in ARM, it is not possible to transfer the user-token to the RP application. Therefore, we couldn’t use this solution.   
   Note: ASC contacted ARM and requested this ability.
2. Through the Portal – in Ibiza portal there is a built-in support to generate user-token to other services. Therefore, we chose this approach.

To generate a user-token through Ibiza portal, we had to add a new resource in the extension manifest, which granted Ibiza the ability to request user-token for Microsoft Defender ATP audience. It takes ~1 week to approve the CR. After the CR is completed we waited 1-2 weeks until they deployed it to PPE and MPAC (ms.portal).  
[Updating the portal manifest in Ibiza](https://msazure.visualstudio.com/One/_git/AzureUX-PortalFx/pullrequest/1885407?_a=files).

To generate token, Microsoft Defender ATP had to pre-authorize our portal application and enable CORs. Pre-authorization is a process which allows adding claims to 1st part applications for other services. It means that we don’t need to ask the user for consent using a pop-up window. Microsoft Defender ATP also pre-authorized our 1st party RP application to perform only GET calls. It is used to validate Microsoft Defender ATP license and enablement before adding Scuba routing rule.

For more information about pre-authorization: [https://identitydocs.azurewebsites.net/static/aad/preauthorization.html](https://nam06.safelinks.protection.outlook.com/?url=https%3A%2F%2Fidentitydocs.azurewebsites.net%2Fstatic%2Faad%2Fpreauthorization.html&data=02%7C01%7CNir.Benjano%40microsoft.com%7C3246e8e929394d594bf208d727641318%7C72f988bf86f141af91ab2d7cd011db47%7C1%7C0%7C637021183095455073&sdata=bMKzKSYAccEFBHGqFRT1bjxumR6AXCIvRbhrpN4v6GY%3D&reserved=0)

To enable CORs, Microsoft Defender ATP had to add the portal domains. We also had to add Microsoft Defender ATP domains to our trusted domains in Sentinel portal configuration.

## Allowed portal domains by Microsoft Defender ATP

* Hosting Service –
  + PPE – <https://hosting.onecloud.azure-test.net>
  + MPAC – <https://ms.hosting.portal.azure.net>
  + PROD – <https://hosting.portal.azure.net>
* Azure Portal –
  + PPE – <https://df.onecloud.azure-test.net> & <https://df.portal.azure.com>
  + PROD – <https://portal.azure.com>
  + MPAC –<https://ms.portal.azure.com>

**It is recommended to allow all portal domains using \*.portal, but since it involved complex modifications to Microsoft Defender ATP code, we postponed it to GA.**

For more information about hosting service domains: <https://github.com/Azure/portaldocs/blob/dev/portal-sdk/generated/top-extensions-hosting-service.md#hosting-service-diagnostics>

## Trusted Microsoft Defender ATP domains by Sentinel

* PPE – Code can be found at: [df.onecloud.azure-test.net.json](https://msazure.visualstudio.com/One/_git/ASI-Portal?path=%2Fsrc%2FASIExtension%2FContent%2FConfig%2Fdf.onecloud.azure-test.net.json&version=GBdevelop).
  + <https://threatintel-eus-stg.cloudapp.net>
* PROD – Code can be found at: [portal.azure.com.json](https://msazure.visualstudio.com/One/_git/ASI-Portal?path=%2Fsrc%2FASIExtension%2FContent%2FConfig%2Fportal.azure.com.json&version=GBdevelop).
  + <https://threatintel-cus.securitycenter.windows.com>
  + <https://threatintel-eus.securitycenter.windows.com>
  + <https://threatintel-neu.securitycenter.windows.com>
  + <https://threatintel-uks.securitycenter.windows.com>
  + <https://threatintel-ukw.securitycenter.windows.com>
  + <https://threatintel-weu.securitycenter.windows.com>



Figure 1 - Private Preview Pipeline



Figure 2 - enable Microsoft Defender ATP from the portal



Figure 3 - add Microsoft Defender ATP Scuba routing rule

## Alert Gateway

Scuba read routing rules from the Scuba cosmos DB, based on those rules, they fetch the data and send it to the relevant consumer. In Sentinel, all security alerts are sent to the Alert Gateway. The Alert Gateway read messages from an input event hub that scuba write to. Each message is an alert which is being process by the Alert Gateway. After processing the message, the alert will be sent to the relevant consumers, such as Log Analytics and Graph store (for investigation).

When adding a new security alert provider, there are several things to remember.

1. Not all providers send the ‘ProductName’ field. This field is being populated using a look up which maps provider name to the relevant product name. The code can be found at: [ProductOverrideEnrichment](https://msazure.visualstudio.com/One/_git/Rome-Detection-AlertGateway?path=%2Fsrc%2FAlertPipeline%2FAlertEnrichment%2FProductOverrideEnrichment.cs&version=GBdevelop). If you don’t update this mapping, a monitor will be triggered and the field ‘ProductName’ might not be populated.
2. Some providers send ‘update’ to existing alerts. Those updates have the same ‘ProviderAlertId’ but with different values.
3. In Sentinel, we use the field ‘SystemAlertId’ as the alert identifier. To support alert ‘update’ we generate the field ‘SystemAlertId’ based on the ‘ProviderAlertId’ and the ‘ProviderName’. This way we will always have a single ‘SystemAlertId’ for each alert. To apply this logic, the alert provider should be whitelisted here: [EventsToSecurityAlertConverter](https://msazure.visualstudio.com/One/_git/Rome-Detection-AlertGateway?path=%2Fsrc%2FAlertPipeline%2FAlertConverter%2FEventsToSecurityAlertConverter.cs&version=GBdevelop&line=118&lineStyle=plain&lineEnd=118&lineStartColumn=21&lineEndColumn=61).
4. Currently, Log Analytics limits the alert size to 32kb.

For more information about the Alert Gateway and sample queries use: [Alert gateway TSG](onenote:https://microsoft.sharepoint.com/teams/AzureSecurityInsights/Shared%20Documents/Notebooks/Book%20of%20Azure%20Security%20Insights/LiveSite/LiveSite.one#Alert%20Gateway%20TSG&section-id=%7B95D8253C-F95A-4084-A824-9FF4245976D6%7D&page-id=%7B8434231A-D98C-4700-8160-1B3536061C2D%7D&end).

## Work in Microsoft Defender ATP

* Create the enablement endpoints.
* Pre-authorize Sentinel portal and RP application.
* Enable CORs for portal endpoints.

## Work in Sentinel

* Implement add MDATP scuba routing rule from RP.
* Implement MDATP connector in the portal.
* Add MDATP trusted domains.
* Enable MDATP token generations from Ibiza portal.
* Implement E2E tests (TIP).
* Update Alert Gateway to support MDATP ‘updates’ and ‘ProductName’.

# Microsoft Defender ATP Enablement pipeline (Public)

Microsoft Defender ATP is a security alert provider. In Sentinel, we allow creating incidents (cases) based on security alerts, this feature is called Alert Filtering which is under [ASI-AnalyticsManagement](https://msazure.visualstudio.com/One/_git/ASI-Analytics-Management). There are special alert rules that generate incidents based on the providers.

To enable alert filtering we need to:

1. Add MDATP provider to the white-listed providers.
   * Reference: MicrosoftSecurityRulesValidator
2. Create MDATP yaml which will be used to create the actual rule.
   * This template will be added to the gallery under Analytics scenario.
   * Reference: [AADIP\_any\_any.yaml](https://msazure.visualstudio.com/One/_git/ASI-Analytics-Management?path=%2Fsrc%2FAnalyticsTemplatesService%2FTemplatesResources%2FAlertFiltering%2FAADIP_any_any.yaml&version=GBdevelop)
3. Update the tests by running them is see what fails.
   * Reference: [TemplatesLoaderAndCreatorTest](https://msazure.visualstudio.com/One/_git/ASI-Analytics-Management?path=%2Ftest%2FAnalyticsTemplatesService.Test%2FTemplatesLoader%2FTemplatesLoaderAndCreatorTest.cs&version=GBdevelop)
4. (Only after 1-3) Add alert filtering instruction to MDATP connector.  
   Reference: [AlertActionLogic.ts](https://msazure.visualstudio.com/One/_git/ASI-Portal?path=%2Fsrc%2FASIExtension%2FClient%2FScenarios%2FDataConnectors%2FSections%2FInstructionsSection%2FActions%2FFilterAlertActionLogic.ts&version=GBdevelop) , [McasDataModel.ts](https://msazure.visualstudio.com/One/_git/ASI-Portal?path=%2Fsrc%2FASIExtension%2FClient%2FData%2FDataConnectorsV3%2FConnectors%2FMcasDataModel.ts&version=GBdevelop&line=107&lineStyle=plain&lineEnd=107&lineStartColumn=21&lineEndColumn=40)

Microsoft Defender ATP alerts can be addressed as a collection of events. Therefore, not all entities will appear on every event. When fetching entities for incidents (cases) in Sentinel, we only take the latest alert and fetch its entities. This approach won’t work any more since the latest alert might have missing entities.

To fix this issue there are 2 solutions:

1. Fetch entities using graph store.  
   Work item : <https://msazure.visualstudio.com/One/_queries/query-edit/06500962-9ef8-4206-8feb-4385358d870b/>
2. Fix the entities query in the portal.
   * Fix the entity count.
   * Fix the entities in entities tab in the single case blade.

Microsoft Defender ATP tenant region might be different from the workspace region. Therefore, we should add the tenant region to the Microsoft Defender ATP Scuba routing rule. It will be used to fetch the tenant alerts from the closest region.

To implement this, there are 2 solutions:

1. Ask Microsoft Defender ATP to add the tenant API polling region to the GET /tenanturl endpoint.
2. Map tenant regional domain to the relevant API polling region.

## Work in Microsoft Defender ATP

* Add tenant region for API polling to the GET /tenanturl endpoint.

## Work in Sentinel

* Support incident (case) creation based on Microsoft Defender ATP provider.
* Fix entities queries in incidents (cases) scenario.
* Add the tenant region for API polling to the Scuba routing rule.
* Security review.

# Microsoft Defender ATP Enablement pipeline (GA)

Microsoft Defender ATP implemented a SIEM integration which is based on API polling. This is the official way to export alerts from Microsoft Defender ATP to consumers.

The alerts from the SIEM integration are received in Microsoft Defender ATP scheme. As Microsoft Defender ATP claim, those alerts are being enriched by additional data.

For more information about SIEM integration: <https://docs.microsoft.com/en-us/windows/security/threat-protection/microsoft-defender-atp/enable-siem-integration>.

Microsoft Defender ATP SIEM integration support 3 endpoints:

* For EU: https://wdatp-alertexporter-eu.windows.com/api/alerts
* For US: https://wdatp-alertexporter-us.windows.com/api/alerts
* For UK: https://wdatp-alertexporter-uk.windows.com/api/alerts

All tenants support all regions. But it is recommended to choose the closest region of the tenant region as defined during the onboarding process to Microsoft Defender ATP, to increase performance (shorter round trip).

Those endpoints receive a time window to retrieve its alerts. For more information about those endpoints visit: <https://docs.microsoft.com/en-us/windows/security/threat-protection/microsoft-defender-atp/pull-alerts-using-rest-api>.

It is important to mention that the 90 percentile of processing alert in Microsoft Defender ATP, is 2 minutes. There is an additional delay for the API polling of 1 minute to prevent concurrency issues.

Usually, customer who wanted to integrate with this solution had to generate a token through Microsoft Defender ATP portal, which expires after 90 days. Since this is not an option for Sentinel, we will request a consent from the user to pull the alerts from this API.

There are 2 options to receive this consent:

1. Use the 1st party application which already have consent.
2. Ask the user to give consent to a different application style Office 365 connector.

Both solutions will work but the 2nd will allow us to connect multiple Microsoft Defender ATP tenants to Sentinel. However, this solution will cause our existing customers who already onboarded Microsoft Defender ATP connector to stop receiving alerts until they give the application permission to fetch them.

### Scale concerns

1. Multiple request for the same tenant – Scuba reads the routing rule and based on them, perform the API polling. There might be several routing rules for the same tenant but for different workspaces.
   1. Now, there are 11,000 workspaces in Sentinel. When Azure Sentinel will be GA, this number will increase.
   2. It is hard to tell how many customers will add Microsoft Defender ATP connector.
2. Too many alerts within the time window – Microsoft Defender ATP supports up to 10,000 alerts per call.
   1. We have large customers that might reach those numbers.
   2. Scuba will try to perform retry in case a time window failed until succeeded.
   3. Scuba will poll new alerts every 5-10 minutes.

How are we going to handle those concerns?

1. Scuba is going to fetch alerts once per tenant for each time window.
2. Scuba is going spread the calls over time during the time window.
3. Scuba will use the tenant region defined in the routing rule, to perform the polling. They will have a default region in case the rule is missing the region.
4. When reaching Microsoft Defender ATP threshold, they will return the latest alerts. We can use it to implement paging over Microsoft Defender ATP alerts to fetch the entire time window.
5. If needed, Microsoft Defender ATP will increase the scale.

## Work in Microsoft Defender ATP

* Support all azure portal domains and hosting service.
* Increase the scale of the SIEM integration API to support large number of requests.
  + They should support at least ~11,000 request per 5-10 minutes.
* Authorize an application to perform the API polling using tenant-token. Sentinel will provide the application. It can be:
  + 1st party application.
  + Designated application.

## Work in Sentinel

* Provide Microsoft Defender ATP the application that will perform the polling.
* Provide Scuba the secret to user Sentinel 1st party application.

## Work in Scuba

* Implement the API polling logic.
  + Support existing rules using a default API polling region.
  + Retry logic to re-fetch alerts of failed time window.
  + Implement paging in case we reach the max number of alerts per call.
  + Monitor unauthorized responses from the API polling. It might indicate that the customer removed the consent.
  + Performance wise implementation to prevent duplicate call to the API polling and to spread the calls over time.



Figure 4 - API polling logic (Scuba)

# Important contacts

Nitzan More Sarid & Itay Fiszman – implemented the Microsoft Defender ATP enablement API for Sentinel.

Lior Carmy – responsible for the SIEM integration in Microsoft Defender ATP.

Nir Benjano– implemented the preview pipeline for Sentinel.

Yonggang Qiao & Matthew Bucher – responsible for the API polling in Scuba.

Yotam Shkedi – responsible for the alert gateway.

Idan Levin, Amit Sharan & Yael Flashner – responsible for the authentication process and decision making in Microsoft Defender ATP.

Michal Ben Yaacov– team leader of the data collection theme in Sentinel.

# Open questions

* Scuba – What will happen if there are too many alerts within the time window. Should MDATP support paging?
* Sentinel – How long does alerts stay in API polling.