Microsoft

Azure Sentinel Technical Playbook for MSSPs

How to deploy Azure Sentinel as a managed security services provider

Published: March-2021, Revision: V0.9

Authors:
Javier Soriano (Senior Program Manager, CxE Sentinel)
Ty Balascio (Senior Program Manager, CxE)
Yaniv Shasha (Senior Program Manager, CxE Sentinel)
Chris Boehm (Senior Program Manager, CxE Sentinel)
Chi Nguyen (Program Manager 2, CxE Sentinel)
Paul Cullimore (Senior Business Strategy Manager)
Edi Lahav (Senior PM Manager, CxE Sentinel)
Richard Diver (Senior Business Strategy Manager)
Waleed Bedair (Senior Program Manager, CxE Partners security)

NOTE: Certain recommendations in this white paper may result in increased data, network, or compute resource usage, and may increase your license or subscription costs.

Introduction

Thank you for considering Azure Sentinel as the heart of your Managed security service providers or SIEM integration practice. Azure Sentinel is a cloud-native security information and event manager (SIEM) platform that uses built-in AI to help analyze large volumes of data across an enterprise—fast. Azure Sentinel aggregates data from all sources, including users, applications, servers, and devices running on-premises or in any cloud, letting you reason over millions of records in a few seconds. It includes built-in connectors for easy onboarding of popular security solutions. Collect data from any source with support for open standard formats like CEF and Syslog.

Target audience

This document informs Microsoft partners researching how to integrate Azure Sentinel into their portfolio of services. It is written through the lens of Implementers & SOC architects who seek a distilled technical walkthrough of:

Azure Sentinel's capabilities
Technical dependencies
Data collection models
Multi-tenant management
Threat detection & analytics
Investigation processes
Strategies for automated response
Activity summaries and reports
Cost models and data storage

Beyond MSSPs, this document aims to guide large organizations and institutions who operate security operations within environments requiring multi-tenant architectures.
This document is relevant both to MSSPs that are new to Sentinel and those who are already experienced with Sentinel Experienced MSSP may want to focus on the following chapters:

- Architecture
- Sizing & Pricing/Cost - Long term storage options summary
- Automation/SOAR - MSSPs design considerations for automation rule and playbooks
- Analytic Rules
- Azure Sentinel Workbooks - Intellectual property protection
- DevOps / CI/CD automation

The Azure Sentinel value for MSSPs

Managed security service providers design business models based on scale and efficiency. The expense incurred by onboarding the first few clients define the templates for future deployments. Operating efficiently requires analyzing each use case required within a customer segment and applying a reusable process with a bias toward automation.

Your customers already understand the importance of employing rigorous security oversight. Introducing any change to tools or processes requires an examination of workload impact, system compatibility, and improved ROI. As compliance requirements, environmental complexity, and threat landscapes expand they rely on you for providing exhaustive security services as well as summaries and outcomes of the threats managed. Azure Sentinel's architecture anticipates these requirements and scales to address them with extensive built-in capabilities complemented with rich extensibility for integrating your existing business processes.

Efficient customer onboarding

Thorough and expedient automated customer onboarding is the 1st area your practice will see value with Azure Sentinel. Closing the time gap between customer commitment and 'fully deployed' enables stronger security for the client and lower costs for your team. Azure Sentinel deploys in minutes using templates customizable for your needs. Likewise, well tested data connectors cover a wide spectrum of popular resources, and a rich library of automation is available for nearly any scenario.

Optimizing system administration costs

Each customer operating environment contains unique characteristics. Tuning is necessary to ensure signals from all resources are collected, network topologies are understood, and monitors are in place. Moreover, resources are in a state of constant change. Azure Sentinel, in conjunction with Azure's governance capabilities, can adjust to these changing conditions programmatically.

Scaling SOC operations

As industries seek to improve efficiency, a larger dependence is placed on digitally transforming business processes. In addition to driving larger data estates, the ever-expanding variety of computing resources deployed result in exponential growth in threat signals requiring analysis. MSSPs differentiate themselves by filtering and distilling these signals down to the most important and most actionable. As a cloud native SIEM, Azure Sentinel's capabilities of threat detection and analytics provides a strong foundation toward this need. Combined with the specialized tuning of your SOC team, your client's signal-to-noise ratio enriches.

Reutilization of intellectual property

From initial deployment through automated threat responses, all MSSPs seek opportunities for re-use across multiple client environments. Deployment automation, detection rule creation and tuning, visualizations, investigation workbooks, and client escalation workflows are but some of the many use cases ripe for scale. Not only will these efficiencies lead to more agility across your client base, but these investments also elevate your staff from repetitious, often mundane tasks.

Your Azure Sentinel journey

Use this guide as a reference for understanding Azure Sentinel's capabilities, planning, and budgeting considerations, and technical resources; formatted from a MSSP's point of view. This document complements official documentation, Certification learning paths, and community resources. Links to those resources provided in the last chapter.

Fundamentals

MS fundamentals

Microsoft's Cloud offerings consist of Microsoft 365, Microsoft Azure and Microsoft Dynamics 365.

While Azure Sentinel supports signals from any workload, in this topic we'll focus mainly on Office 365 and Microsoft Azure cloud security offerings. It's important to understand the hierarchy of the major entities of these cloud offerings as it has a significant impact on the architecture of MSSP deployment.

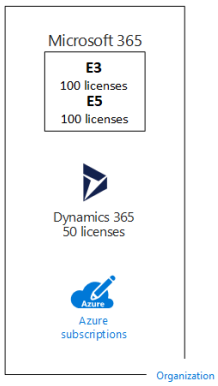
Microsoft provides a hierarchy of Organizations, Tenants, Subscriptions, and User accounts for consistency of identities and billing across the cloud offerings.

Following are hierarchy elements:

- **Organization -** represents a business entity that is using Microsoft cloud offerings, typically identified by one or more public Domain Name System (DNS) domain names, such as contoso.com. The organization is a container for subscriptions.
- **Subscriptions –** A subscription is an agreement with Microsoft to use one or more Microsoft cloud platforms or services, for which charges accrue based on either a per-user license fee or on cloud-based resource consumption. Organizations can have multiple subscriptions for Microsoft's cloud offerings.
- **User accounts -** User accounts for all of Microsoft's cloud offerings are stored in an Azure Active Directory (Azure AD) tenant, which contains user accounts and groups. An Azure AD tenant can be synchronized with an existing on-prem Active Directory Domain Services (AD DS) accounts using Azure AD Connect, a Windows server-based service. This is known as directory synchronization.

  Following is an example of an Azure AD Tenant that contains user accounts and groups for various Microsoft's Cloud Services
- **Tenants –** For SaaS cloud offerings, the tenant is the regional location that houses the servers providing cloud services. For example, the Contoso Corporation chose the European region to host its Microsoft 365, EMS, and Dynamics 365 tenants for the 15,000 workers in their Paris headquarters. Azure PaaS services and virtual machine-based workloads hosted in Azure IaaS can have tenancy in any Azure datacenter across the world. You specify the Azure datacenter, known as the location, when you create the Azure PaaS app or service or element of an IaaS workload.

  An **Azure AD tenant** is a specific instance of Azure AD containing accounts and groups. Paid or trial subscriptions of Microsoft 365 or Dynamics 365 include a free Azure AD tenant. This Azure AD tenant does not include other Azure services and is not the same as an Azure trial or paid subscription.
- **Licenses -** For Microsoft's SaaS cloud offerings, a license allows a specific user account to use the services of the cloud offering. You are charged a fixed monthly fee as part of your subscription. Administrators assign licenses to individual user accounts in the subscription. For the example in Figure 2, the Contoso Corporation has a Microsoft 365 E5 subscription with 100 licenses, which allows to up to 100 individual user accounts to use Microsoft 365 E5 features and services.

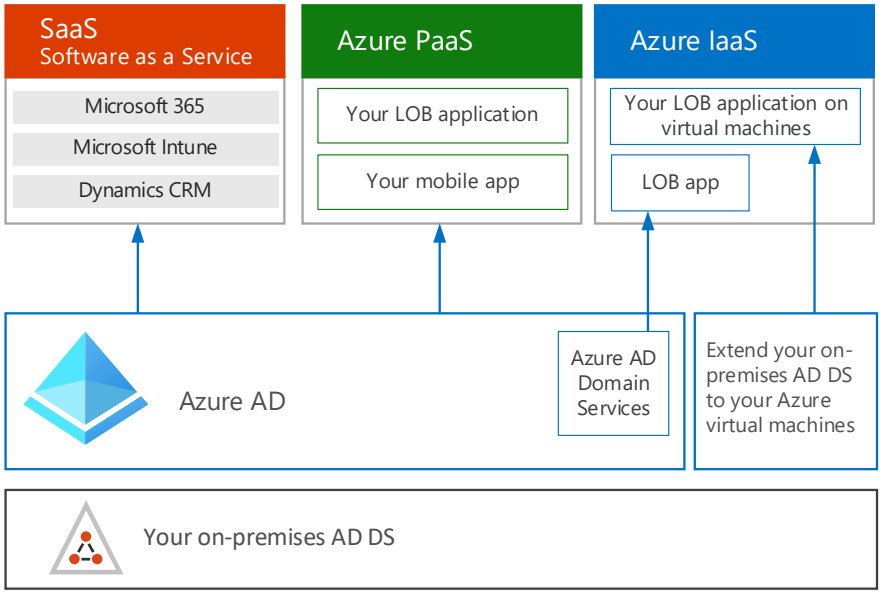


Connecting it all together

- An organization can have multiple subscriptions
- A subscription can have multiple licenses
- O365 licenses can be assigned to individual user accounts
- User accounts are stored in an Azure AD tenant

  Multiple Microsoft cloud offering subscriptions can use the same Azure AD tenant that acts as a common identity provider. A central Azure AD tenant that contains the synchronized accounts of your on-premises AD DS provides cloud-based Identity as a Service (IDaaS) for your organization.

  In the following diagram the common Azure AD tenant used by Microsoft's SaaS cloud offerings, Azure PaaS apps, and virtual machines in Azure IaaS that use Azure AD Domain Services. Azure AD Connect synchronizes the on-premises AD DS forest with the Azure AD tenant.



Relevant resources:

[Microsoft's hierarchy for Cloud Offerings]

[Microsoft Cloud Identity for Enterprise Architects]

[Microsoft 365 identity models and Azure Active Directory]

[Quickstart: Setup a tenant]

Azure fundamentals

The Azure Cloud platform consists of more than 200 products and Cloud Services. It allows customers to build and run their solutions across multiple clouds, on-premise and at the edge with a variety of tools and frameworks.

Azure Service's are available globally and partners can choose the best region for their customers' needs based on technical and regulatory considerations: service capabilities, data residency, compliance requirements, and latency.

A **Region** is a set of datacenters deployed within a latency-defined perimeter and connected through a dedicated regional low-latency network. Azure gives you the flexibility to deploy applications where you need to, including across multiple regions to deliver cross-region resiliency.

An **Availability Zone** is a high availability offering that protects your applications and data from datacenter failures. Availability Zones are unique physical locations within an Azure region. Each zone is made up of one or more datacenters equipped with independent power, cooling, and networking.

To ensure resiliency, there's a minimum of three separate zones in all enabled regions. The physical separation of Availability Zones within a region protects applications and data from datacenter failures. Zone-redundant services replicate your applications and data across Availability Zones to protect from single-points-of-failure.

To achieve comprehensive business continuity on Azure, build your application architecture using the combination of Availability Zones with Azure region pairs. You can synchronously replicate your applications and data using Availability Zones within an Azure region for high-availability and asynchronously replicate across Azure regions for disaster recovery protection.

Azure resources reside in **Subscriptions** and each resource is assigned to a **Resource Group**. For cases where many subscriptions exist a more efficient way is required manage access, policies, and compliance for those subscriptions.

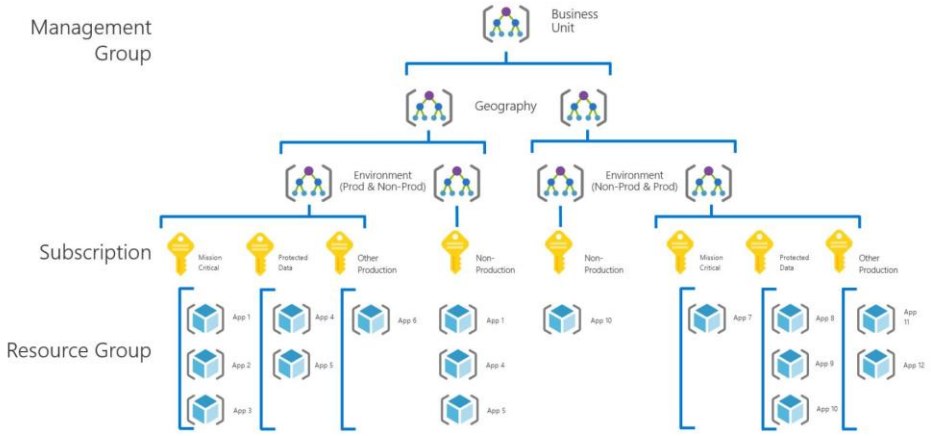
A **Resource Group** is a container that holds related resources for an Azure solution. The resource group includes those resources that should be manage as a group. The decision which resources belong in a resource group based on what makes the most sense for the customer.

Azure Resource Manager (ARM) - is the deployment and management service for Azure. It provides a management layer that enables you to create, update, and delete resources in your Azure account. It includes management features, like access control, locks, and tags, to secure and organize the resources after deployment.

Azure management groups provide a level of scope above subscriptions. You organize subscriptions into containers called "management groups" and apply your governance conditions to the management groups. All subscriptions within a management group automatically inherit the conditions applied to the management group. Management groups give you enterprise-grade management at a large scale no matter what type of subscriptions you might have. All subscriptions within a single management group must trust the same Azure Active Directory tenant.

Azure Policies help to enforce organizational standards and to assess compliance at-scale. It also helps to bring your resources to compliance through bulk remediation for existing resources and automatic remediation for new resources. Common use cases for Azure Policy include implementing governance for resource consistency, regulatory compliance, security, cost, and management. Policy definitions for these common use cases are already available in your Azure environment as built-ins to help you get started.

The following diagram shows an example of creating a hierarchy for governance using management groups.



Relevant resources:

Azure Security best practices

Microsoft Cloud Adoption Framework for Azure

Regions and Availability Zones in Azure

Overview of the reliability pillar

Products available by region

What are Management Groups?

What is Azure Policy?

What is a Resource Manager?

What is Azure role-based access control (RBAC)?

Log Analytics fundamentals

Log Analytics is a tool in the Azure portal to edit and run log queries from data collected from various data sources and interactively analyze their results. Log Analytics queries can be used to retrieve records matching particular criteria, identify trends, analyze patterns, and provide a variety of insights into your data.
Azure Monitor, and its Log Analytics module, is the underlying log management platform powering Azure Sentinel. As such, any source that sends logs to Azure Monitor or Log Analytics inherently supports Azure Sentinel. As soon as Azure Sentinel is enabled for a Log Analytics workspace all collected sources are available at Azure Sentinel for further analysis, hunting, threat mitigation and additional actions taken by the SOC analyst.

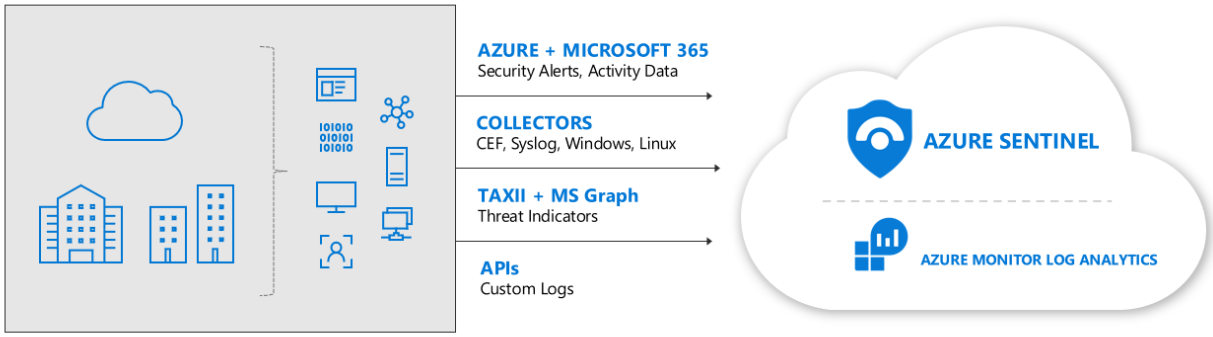
Data collected by Azure Monitor Logs is stored in one or more **Log Analytics workspaces**. The workspace defines the geographic location of the data, access rights defining which users can access data, and configuration settings such as the pricing tier and data retention. At least one workspace should be created to use Azure Monitor Logs. A single workspace may be sufficient for all of your monitoring data, or you may choose to create multiple workspaces depending on the requirements. For example, a requirement to split operational logs and security logs would result in 2 separate workspaces. The security logs workspace will be Azure Sentinel enabled.
Data is retrieved from a Log Analytics workspace using a log query which is a read-only request to process data and return results. Log queries are written in **Kusto Query Language (KQL)**, which is the same query language used by Azure Data Explorer.

The Azure Log Analytics agent or Microsoft Monitoring agent (MMA) collects telemetry from Windows and Linux virtual machines in any cloud or on-premises machines and sends the collected data to your Log Analytics workspace in Azure Monitor and/or Azure Sentinel.

MMA collects the following data sources: Windows Events, Syslog, Performance, IIS logs and Custom logs. Log Analytics workspace configuration allows to decide which data sources will be collected by all connected agents. The **Azure Monitor Agent (AMA)** (Preview) provides new capabilities such as Windows Events filtering, multi-homing support for Linux, improved manageability of collection settings with DCR (Data Collection Rules) and more. For more information about AMA, please refer to the technical docs: https://docs.microsoft.com/en-us/azure/azure-monitor/agents/azure-monitor-agent-overview

Data Destinations - The Windows agent can be multihomed to send data to multiple workspaces and System Center Operations Manager management groups. The Linux agent can send to only a single destination, either a workspace or management group.

In the following example various Connectors and log collection methods are listed to ingest logs from on-premise and/or Cloud Services such as: Office365, Azure, AWS, GCP and more



Data Collection in depth will be covered in the Data collection chapter.

Relevant resources:

Log Analytics Tutorial
Azure Monitor Logs overview
Overview of Log Analytics in Azure Monitor
Log Analytics agent overview
Quickstart: On-board Azure Sentinel
Kusto Query Language (KQL)
Managing and maintaining the Log Analytics agent

Architecture

This section focuses on the high-level design principles that need to be taken into account to provide an Azure Sentinel service as a Managed Security Services Provider (MSSP).

From now on in the document, we will describe the different concepts with one fictitious MSSP called Fabrikam, and two fictitious customers called Contoso and Wingtip.

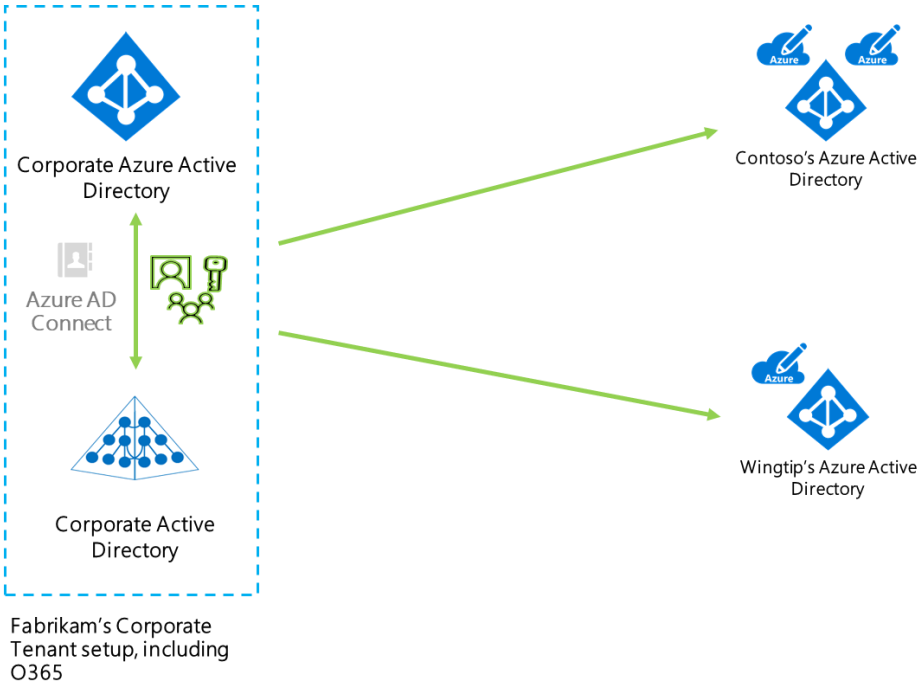
Azure AD tenant topologies

In this section we will explain the different options available to structure your Azure AD tenants to provide an Azure Sentinel service to your end customers. There are fundamentally two options:

- Use a single identity for the MSSP internal services and applications and Azure management services.
- Use separate identities, one for MSSP internal services and applications, and a separate identity to manage your customers.

Single identity model

In this approach, the MSSP has a single Azure AD that is used for internal operations and to serve customers. The following diagram describes this model:



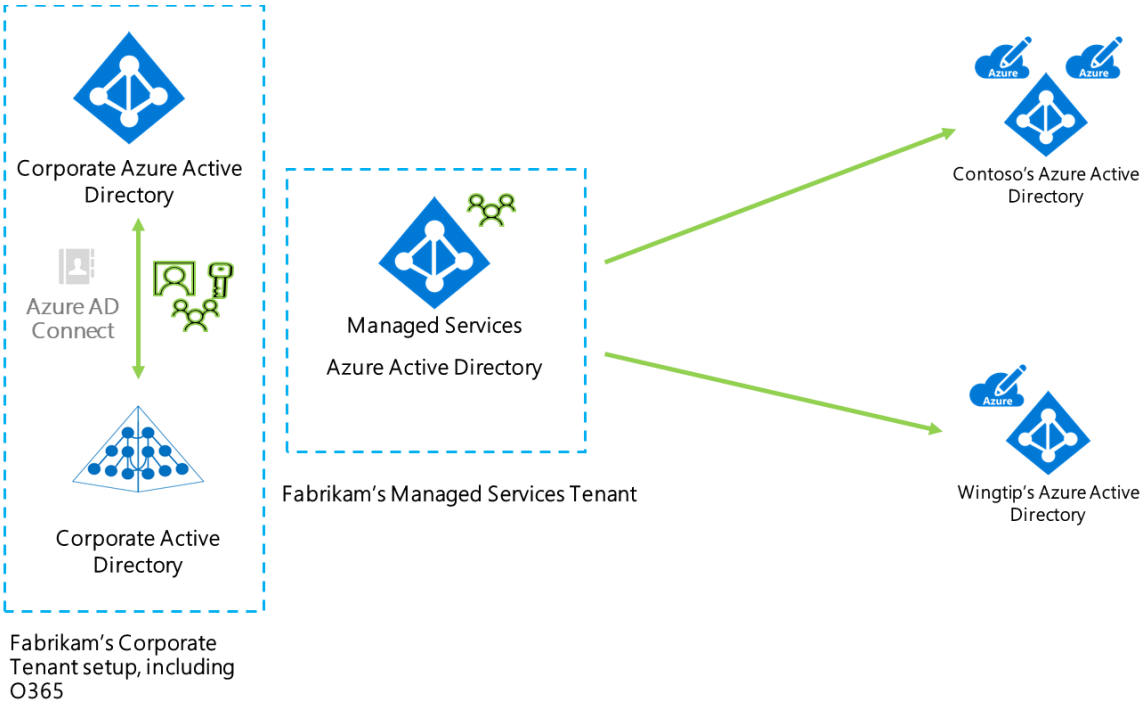
For example, user User1 from Fabrikam's managed services team would have a single identity and set of credentials that he will use to access internal applications like Office365 and also to access the customer environments that Fabrikam manages (we will discuss below what methods are available to access customer environments).

Pros: the great advantage of this model is that you only need to manage one identity for your users. That greatly simplifies your identity management processes and you also have a smaller identity footprint to secure. In our example, User1 will need to change his password just once whenever it expires and if he's terminated, Fabrikam will only have to disable one account.

Cons: the main disadvantage of this model is that you can't separate how your identities are secured by your internal IT to how you want to secure them for your end customer operations. For example, imagine that your internal IT policies require you to use MFA only when logging from outside your office, but your customer forces you to use MFA for each single login operation…this might force your internal IT team to implement new policies that they were not planning.

Multiple identities model

In this approach, the MSSP has two Azure AD tenants, one for its internal services and applications and a separate one to manage Azure customers. The following diagram describes this model:



For example, user User1 from Fabrikam's managed service team, would have two identities, one to use his internal applications like email and others, and an additional identity that he will use when he needs to access customer environments (we will see in a later section how to access customer environments).

Pros: having two separate identities makes it easier for Fabrikam to adapt to different requirements from internal IT and from customers. For example, if Contoso requires Fabrikam to always use MFA when logging, they can just implement this policy in the managed services tenant and leave the corporate tenant unchanged.

Cons: Fabrikam needs to maintain multiple identities for users in the managed services team. Also, User1 will have to remember a separate set of credentials. If an employee is terminated, there needs to be a process in place to remove all related identities for the user.

Continue to the next section to understand how MSSP users can access customer environments.

Accessing the customer environment

Azure Lighthouse

Azure Lighthouse enables cross-tenant management, allowing for higher automation, scalability, and enhanced governance across resources and tenants.
This is the preferred method to access your customer environment because it allows you to manage customer resources as if they were in your own Azure AD tenant.

Azure Lighthouse is based on delegations. Each delegation contains three things: Identities, Roles and Scope.

- **Identities**: These are the identities (normally from the MSSP tenant) that will have access to customer resources. You can specify users, groups or service principals as the recipients of a delegation.
- **Roles**: these are the permissions that the identities will have when accessing customer resources. The roles that can be used here are all Azure Built-in roles with three exceptions captured here. Custom roles are currently not supported. Also, you cannot grant Azure AD roles. See differences between Azure and Azure AD roles here.
- **Scope**: this indicates where the delegation will apply, valid scopes are *subscription* and *resource group*.

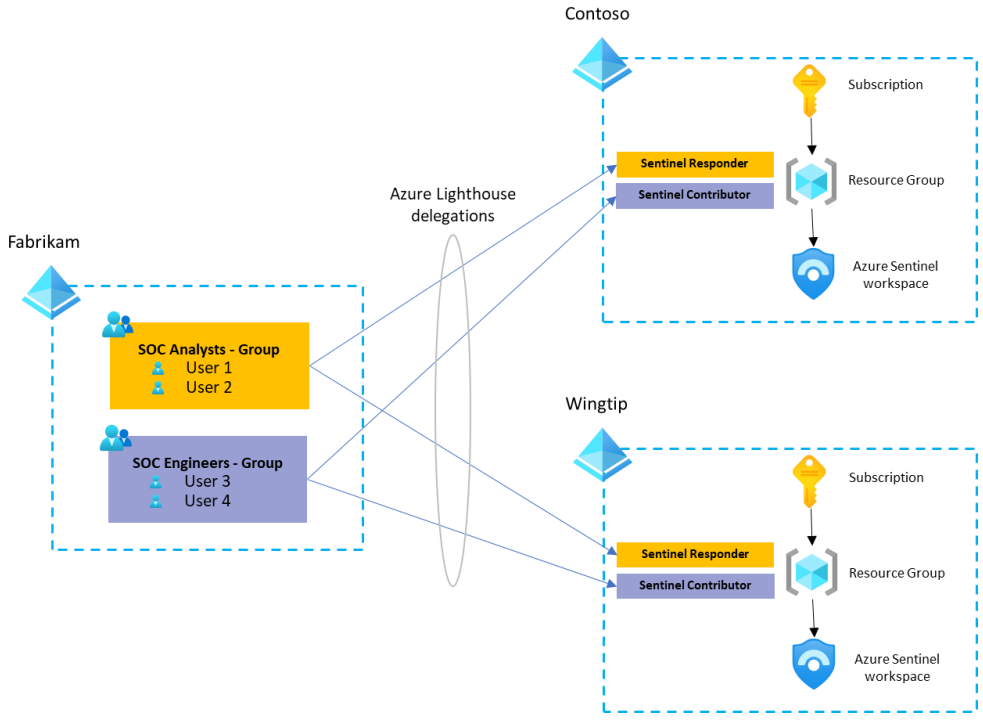
An example delegation would be something like:

- **Identity**: User1 (user), *SecOps* (group) and *automation_spn* (service principal), all from Fabrikam Azure AD tenant
- **Roles**: *Azure Sentinel Contributor, Logic App Contributor*
- **Scope**: resource group *security_rg*

As you can see a single delegation can contain multiple identities and scopes. It can even contain multiple resource groups, but not multiple subscriptions.

There's two different ways to onboard a customer into your Lighthouse management: ARM template or Marketplace offer. Visit this article to get started.

In the context of Azure Sentinel, Azure Lighthouse can be used to manage the service across multiple customers. This is a high-level view of the setup:



As you can see, in this case, the MSSP (Fabrikam) has two delegations for each customer, one for Engineers with an Azure Sentinel Contributor role and one for Analysts with Azure Sentinel Responder role, all of them with delegated access at the resource group level where Azure Sentinel is located. This will effectively provide them with access to the whole resource group with the permissions included in the granted role.

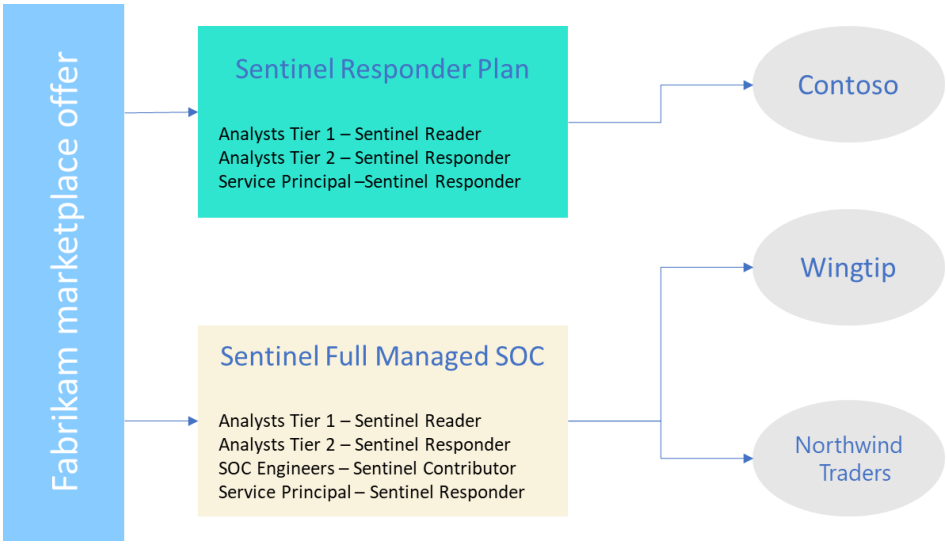
Azure Lighthouse Onboarding

As already mentioned, there are two options: ARM template or Azure Marketplace offer, being the latter preferred as it provides a very easy experience for end customers.

***NOTE** - there are some requirements before an MSSP can publish into the Azure Marketplace. The MSSP must have a silver or gold cloud platform competency level or be an Azure Expert MSP*.

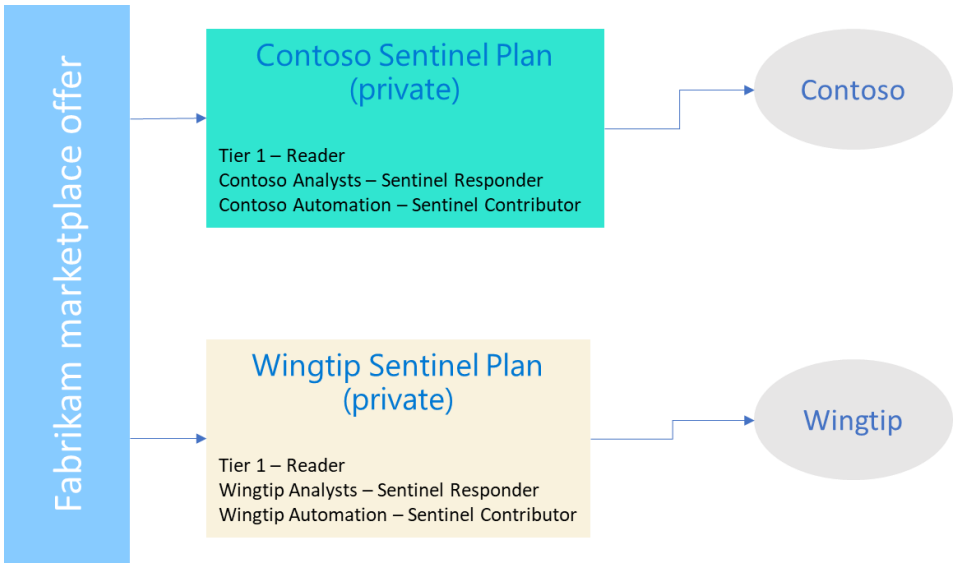
Marketplace offers have an additional concept called **Plan**. A plan defines the service that you will provide to your customer. For example, you can have a marketplace offer to provide managed services for your customers, and within that offer several plans with different flavor: monitoring, backup and recovery, compliance, and full managed service.

In the Azure Sentinel context, you could have a Marketplace offer like this one:



As you can see, inside each plan, you define the groups of users from your tenant that will have access to the customer environment and the permissions that will apply. The plan also includes the scope (resource group of subscription), but we omitted it for simplicity.

You can make these plans **public**, so everyone in Azure can see them, **or private**, if you only want a subset of customer to have access. This would allow to create plans targeted just to specific audiences, like a particular customer or a vertical. Here is an example:



Azure Lighthouse integration with Azure AD Privileged Identity Management (PIM)

Azure Lighthouse has also recently added the possibility to integrate with Azure AD Privileged Identity Management (PIM) (still in preview). This lets you grant delegated permissions to customer tenants on a just-in-time basis so that users only have those permissions for a set duration at a time. Samples and instructions for this integration can be found here.

Azure Lighthouse for Cloud Solution Providers

If you're a Cloud Solution Provider partner, a group of users from the MSSP tenant will automatically get Owner permissions for each customer Azure subscription. In this article, you will find specific guidance on how to combine this with Azure Lighthouse: Cloud Solution Provider program considerations - Azure Lighthouse | Microsoft Docs

Azure AD B2B

Azure Lighthouse is a very important service to get access to customer Azure resources, but it doesn't work for other workloads outside of Azure like Office 365 or Microsoft 365 security services. So, how do MSSPs manage the customer services based on Office 365 or Microsoft 365? Azure B2B is the answer.

Azure Active Directory (Azure AD) business-to-business (B2B) collaboration is a feature within External Identities that lets you invite guest users to collaborate with your organization.

So, the **MSSP users can be "invited" to the customer tenant** to perform management activities in that tenant; MSSP users will appear as *guest* users in the customer tenant. These guests can be then granted roles in the customer Azure AD tenant. The main difference with Lighthouse, is that the guest users can be granted any Azure role (even custom ones) or Azure AD roles (remember that Azure Lighthouse can only grant Azure built-in roles). You can see a detailed discussion about the differences between Azure AD and Azure roles here.

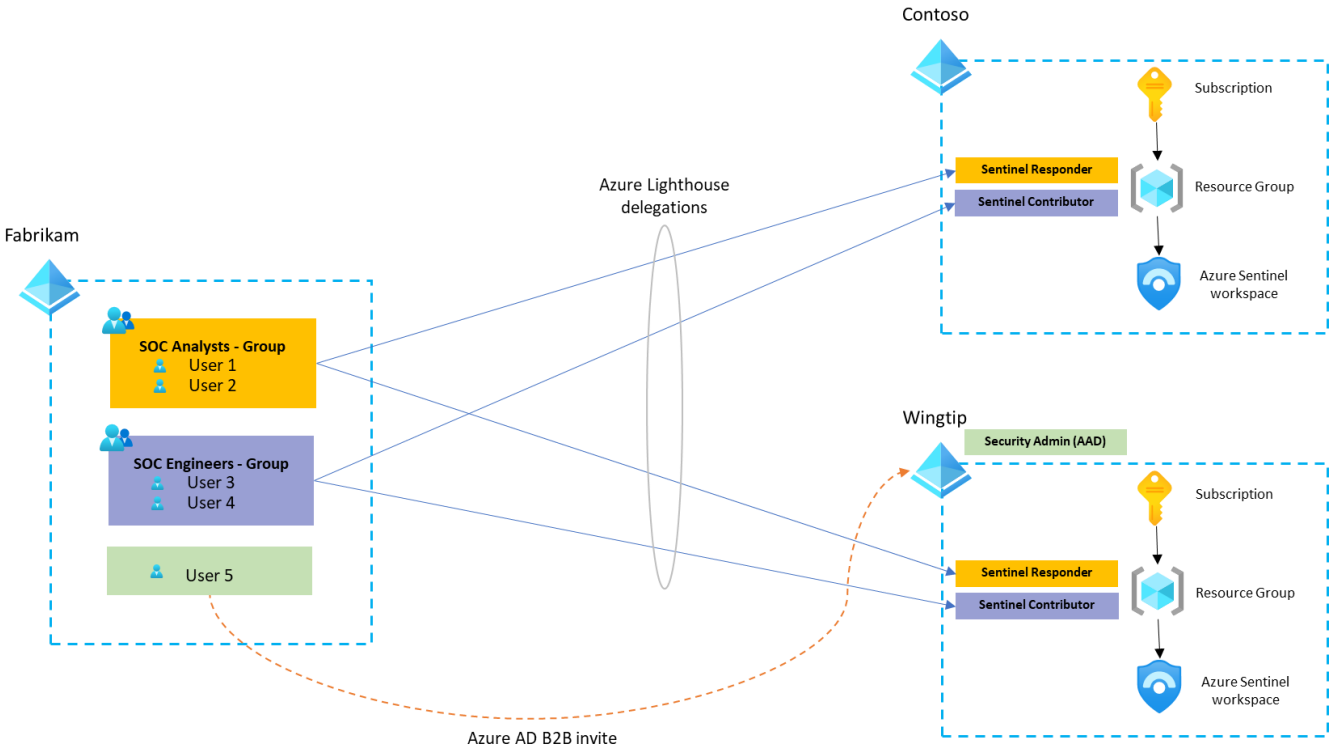
The ability to grant Azure AD roles opens new possibilities like managing Office 365 and Microsoft 365 services.

Then, why do we need Azure Lighthouse?

The Azure B2B approach also has some disadvantages:

- **No cross-tenant management or visibility**. As you are invited into a customer tenant, you have to log into the customer tenant in order to see its resources. This blocks your cross-tenant visibility, as you cannot query multiple tenants at the same time. You would have to log in to one customer, manage that customer, log out, and then go to the next customer, which would be very cumbersome.
- **No ability to invite groups**. Azure B2B are done on a user-by-user basis, you cannot invite an entire group. This brings challenges as you need to manage the lifecycle of each account in multiple places (this limitation can be removed by using Azure Entitlement Management, which we review below).

Taking these disadvantages into account, if you as an MSSP need to manage both Azure and Office 365/Microsoft 365 workloads, **the best approach is to use a combination of Azure Lighthouse and Azure AD B2B invites** (see section below for instructions on how to automate)
This would look like this:

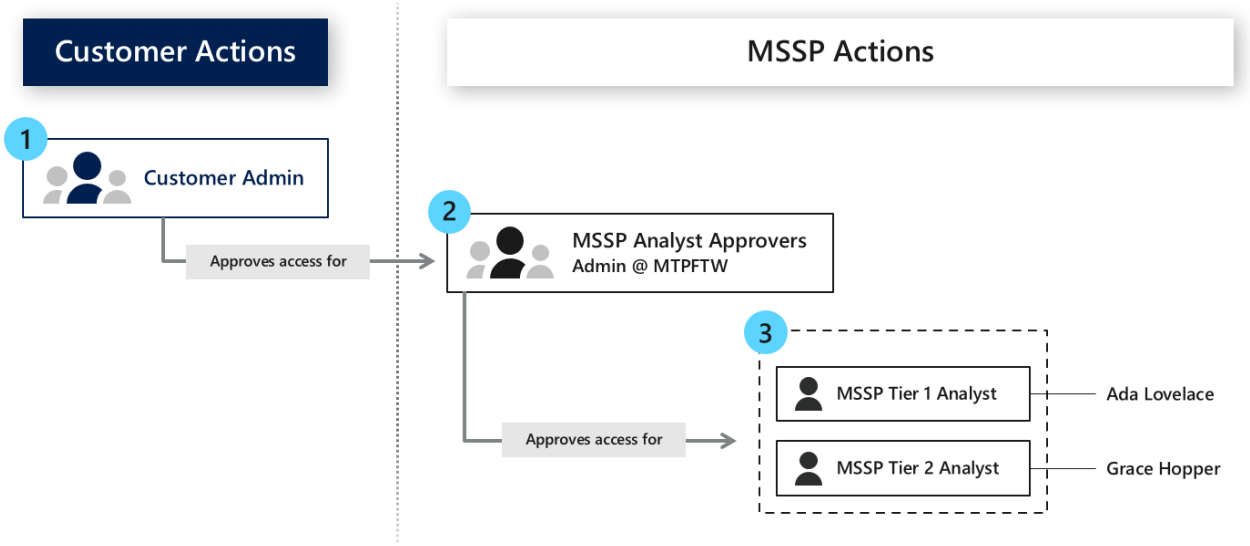


In this picture, we have a new user from Fabrikam (MSSP) that has been invited to Wingtip's Azure AD and is now a *guest* user in that AAD. It has also been granted the Security Admin role. Notice that Security Admin is an Azure AD role, so it can be granted to guest users, but it cannot be granted to users that have delegated access via Azure Lighthouse (remember that Azure Lighthouse can only grant Azure roles). Although not showed in the picture, the same user can have Azure roles delegated through Azure Lighthouse and also be invited as a guest and be granted Azure AD roles like Security Admin or Global Admin. It all depends on how you separate your teams internally.

Azure AD Entitlement Management

Azure Active Directory (Azure AD) entitlement management is an identity governance feature that enables organizations to manage identity and access lifecycle at scale, by automating access request workflows, access assignments, reviews, and expiration. **This feature requires Azure AD P2 license**.

This feature can also be used to manage access from external Azure AD organizations, so it's a **perfect fit for the MSSP access needs when it comes to Microsoft 365 Defender workloads**. This is the high-level architecture for this access model:



With this setup, MSSP users will be automatically invited into the customer tenant (after the appropriate approvals) to manage customer services. You can also assign which specific roles will be granted to those users; these roles should be specially crafted to manage Microsoft 365 Defender workloads.

For an in depth explanation on how to setup Entitlement Management for an MSSP to access customer environments, **read** **this article**. Remember, this is only needed to manage the M365 Defender part of the customer environment, the Azure Sentinel part will be managed through Azure Lighthouse as explained above.

Multi-Workspace design principles

A workspace is an element that makes up part of the Azure Monitor service. Azure Monitor allows for the collection and analysis of two types of data:

- Metric data – numerical values, typically performance data
- Log data – stores text values

Log data is viewed using Azure Log Analytics, with the actual data being stored within a Log Analytics Workspace. Once a workspace has been created within a Resource Group, within a Subscription (within an Azure Tenant) then an Azure Sentinel instance can be created on top of the chosen workspace.

The default workspace design decision is, to keep it simple, with a 'new' single Log Analytics Workspace for Azure Sentinel. However, there are some valid reasons why more than one workspace would be required. These include:

New or reuse?

Many customers and partners already use Log Analytics to store and analyze data from a wide variety of data sources that have little to do with security. Many collect performance indicators and events from servers to monitor application use and operating system behavior. It might be tempting to simple reuse this workspace for Azure Sentinel as well. The Azure Sentinel solution applies to the entire workspace. If a workspace is ingesting 500GB daily, but only about 50GB of this is interesting to Azure Sentinel, Azure Sentinel will still charge for the full 500GB. There is no way of carving up a workspace so that Azure Sentinel focuses on specific tables only.

Therefore, the advice is to create a new workspace for Azure Sentinel if a current workspace is already heavily used. In some cases, the non-security log data can be very small and, in this case, there could be an argument to reuse a current workspace instance. The default approach remains, create a new workspace for Azure Sentinel.

Sovereignty

Workspaces are created within an Azure region. An Azure region is a collection (typically a pair) of datacenters grouped around a geographic area which could be a country, an area within a country of even a continent. The full list of where Log Analytics can be run, and therefore a workspace created, can be found here: Azure Products by Region | Microsoft Azure

Some customers or partners may require the data that Azure Sentinel collects to be stored within a specific country. There are currently only a handful of countries that support full data residency. These are the US, Australia, Canada, Japan, and the UK. More are planned for 2021. Note, although Log Analytic workspaces are available in far more regions (in fact, almost all regions), Azure Sentinel which also holds data around how its configured is only stored locally in the countries described above.

Note, moving forward, if possible, create the multiple workspaces within the same Log Analytics 'cluster'. There are roadmap items that address scalability when managing multiple workspaces and clusters is part of the response, see the paragraph below.

MSSPs

The assumption is that MSSPs will almost certainly have to manage multiple workspaces. Microsoft cloud contracts effectively mean that customers data must always be available to them and be exportable to other platforms. Therefore, every customer MUST run their own Azure Sentinel within their own Azure Tenant. There are technical reasons for this as well, as many of the more feature rich data connectors only work within a tenant.

As with Sovereignty above, the advice if possible, is to create new customer workspaces within a single cluster (or indeed multiple clusters for Sovereignty).

Regulatory compliance + others

Larger or complex customers may require additional workspaces as different business groups are subjects to different laws and compliance rules. Remember, when examining these areas that Azure Sentinel operates at two levels, Log Analytics for storing ingested data and houses the workspace but there is a smaller data store which stores configuration data about the Azure Sentinel instance. This section is only concerned with workspaces. Regional versions of Azure Sentinel are covered under the Sovereignty section above.

Azure Active Directory tenant boundary

A solid reason for additional workspaces is more technically focused. Many customers will have multiple Azure Tenants which are defined by an instance of Azure Active Directory. Many of the more feature rich direct connectors that ingest data into Azure Sentinel only work within an Azure Tenant boundary. These include the incoming Microsoft 365 Defender 2-way connector that can sync incidents between the two platforms as one example. There are approximately 20 other connectors that work within a tenant.

Access control

This can typically be achieved using RBAC as described in the previous section. History has taught us that adding more complexity to enforce access control rarely works and is why many larger enterprises have so many on-premises Active Directory domains, many of which are poorly managed!

Split billing

Again, not normally a good reason for splitting out workspaces. Billing can be configured to be quite granular to show costs per workspace, resource group or subscription as needed.

Log Analytic clusters

Log Analytic Dedicated Clusters are collections of workspaces that must be CREATED within the same Azure region. Together these workspaces must have a minimum of 1TB daily ingestions to qualify. In addition, only two clusters can be created per region and subscription. Though each cluster can hold up to 1000 workspaces.

Please note, a current workspace cannot be moved into a cluster, it must be created within the target cluster. Therefore, a migration strategy would be needed for say an MSSP adopting multiple customers already running Azure Sentinel. Migration tooling is on the roadmap, but no dates have been assigned!

Whilst not a reason to create additional workspaces, clusters do allow for greater use of multiple workspaces as the limits mentioned in the section below will be extended in the future partly due to using Log Analytic clusters. Therefore, it would be wise for an MSSP thinking of onboarding multiple customers to think future flexibility and start to investigate clusters now.

Limits

There are, however, some limits to managing multiple workspaces which are being addressed on the roadmap but exist today.

There is a limit of 10 workspaces than can be selected on the cross-incident view within the Azure Sentinel UI. This limit is expected to be raised during 2021, part of this will be down to clustering (see previous section). Note Azure Lighthouse is needed for an MSSP to view multiple workspaces at once.

There is a limit of 100 workspaces that can be queried by a single KQL query. Though, in practice, the performance of a query running across 100 workspaces will be severely impacted by where these workspaces exist and any network latency which will slow down the query. Again, clustering will help with this area of performance.

A factor to consider when constructing queries across multiple workspaces is that common queries would need to be updated each time an additional workspace was added. To help avoid this admin complexity, the query that contains the workspace names to reference, can be saved as a KQL function and then just use an alias to reference that function (more details here).

An overview of the current limits is in the table below:

Type	Workspace Limit	Description
Cross workspace Incident view	10	Up to 10 workspace instances (e.g., customers) can be displayed at one time.
Cross workspace query	100	A KQL query can contain a workspace reference, well 100 of them though other performance bottlenecks (e.g., network latency) may restrict this max.
Cross workspace workbooks	100	Applies same limit as above
Cross workspace Analytic Rules	20	You can include up to 20 workspaces in a single analytics rule. See Analytics Rule section for more details.
Max analytic rules per workspace	512	

Moving a workspace

It is possible to move workspaces between different tenants within the same Azure region. And it's also possible to move workspaces between subscriptions within the same tenant and the connected ingestion agents will remain connected.

However, not all workspaces are created equal. Azure Sentinel workspaces are special as lots of different analytics including ML algorithms run across the data stored in their workspaces. Thus, today an Azure Sentinel workspace CANNOT be moved. Whilst technically the Azure Log Analytics provides an interface allowing the move, it is not a supported operation for Azure Sentinel. Should a move be necessary, a new instance of Azure Sentinel must be established.

Role Based Access Control (RBAC)

This page in the Azure Sentinel official documentation contains detailed info about the different Azure built-in roles specific to Azure Sentinel. The same page, in the additional roles and permissions paragraph, talks about other roles that are not part of the **SecurityInsights** provider, but that might be needed to operate your Azure Sentinel environment. Read the full article referenced above carefully to understand the roles that you might need as part of your deployment.

Combining the Azure Sentinel roles and Azure Lighthouse, you should plan what permissions will be needed by the different MSSP teams operating your customers Azure Sentinel environment. Here we show a sample table of permissions that can be leveraged as a (simplified) starting point:

Group	Role	Scope	Notes
Security Analysts	Azure Sentinel Responder	Azure Sentinel's Resource Group	View data, incidents, workbooks, and other Azure Sentinel resources. Manage incidents (assign, dismiss, etc.)
	Logic Apps Contributor	Azure Sentinel's Resource Group (or RG where Playbooks are stored)	Attach Playbooks to Analytics Rules and Automation Rules. Run Playbooks. NOTE: this will also allow playbook modification by analysts
Security Engineers	Azure Sentinel Contributor	Sentinel's Resource Group	View data, incidents, workbooks, and other Azure Sentinel resources. Manage incidents (assign, dismiss, etc.). Create and edit workbooks, analytics rules, and other Azure Sentinel resources.
	Logic Apps Contributor	Azure Sentinel's Resource Group (or RG where Playbooks are stored)	Attach Playbooks to Analytics Rules and Automation Rules. Run and Modify Playbooks. NOTE: this will also allow playbook modification by analysts.
Service Principal	Azure Sentinel Contributor	Azure Sentinel's Resource Group	Automated configuration management tasks

This approach can be further enhanced to account for MSSPs with additional complexity and/or bigger scale. For example, we could have separate groups of Security Analysts by customer, vertical (FSI, retail, health) or technical expertise (identity, network, etc.). Let's see an example where we separate by expertise:

Group	Role	Scope	Notes
Security Analysts	Azure Sentinel Responder	Azure Sentinel's Resource Group	View data, incidents, workbooks, and other Azure Sentinel resources. Manage incidents (assign, dismiss, etc)
	Logic Apps Contributor	Azure Sentinel's Resource Group (or RG where Playbooks are stored)	Run Playbooks. NOTE: this will also allow playbook modification by analysts
Identity Engineers	Managed Identity Contributor	Subscription	Create and managed user assigned identities
Networking Engineers	Network Contributor	Subscription	Create and manage networks
Security Engineers	Security Admin	Subscription	View and manage Azure Security Center (policies, rules, alerts) and view log analytics data
	Azure Sentinel Contributor	Azure Sentinel's Resource Group	View data, incidents, workbooks, and other Azure Sentinel resources. Manage incidents (assign, dismiss, etc). Create and edit workbooks, analytics rules, and other Azure Sentinel resources.
	Logic Apps Contributor	Azure Sentinel's Resource Group (or RG where Playbooks are stored)	Run Playbooks. NOTE: this will also allow playbook modification by analysts

In addition to Azure roles, and as explained in the Azure Lighthouse section above, we might need to add Azure Active Directory roles that might be required by security operations teams to manage Microsoft 365 workloads.

What can't you do through Lighthouse?

As we have explained before, the MSSP access to the customer's Azure Sentinel environment will be utilizing Azure Lighthouse. However, there are things that you won't be able to do with just Azure Lighthouse:

- **Onboard some connectors** that require Security Admin or Global Admin permissions in the customer Azure AD tenant. Most of the Microsoft 1st party connectors require one of these permissions to enabled and these are not available through Azure Lighthouse.
- **Assign incidents to a user in the customer Azure Active Directory**. As you manage incidents in the customer workspaces, you will only be able to assign them to users in your own tenant.

You can implement these capabilities by using Azure B2B invites, so a user in the MSSP tenant can have a guest user in the customer AAD tenant with the appropriate permissions to perform these actions.

Data RBAC in Hybrid model

There are two modes in which you can access data within an Azure Sentinel workspace: workspace context or resource context. The following table summarizes both:

	Workspace-context	**Resource-context**
Who is each model intended for?	Central administration. Administrators who need to configure data collection and users who need access to a wide variety of resources.	Application teams. Administrators of Azure resources being monitored.
What does a user require to view logs?	Permissions to the workspace.	Read access to the resource. Permissions can be inherited or directly assigned to the resource
What is the scope of permissions?	Workspace. Users with access to the workspace can query all logs in the workspace from tables that they have permissions to	Azure resource. User can query logs for specific resources, resource groups, or subscription they have access to from any workspace but can't query logs for other resources.
How can user access logs?	- Start Logs from Azure Monitor menu. - Start Logs from Log Analytics workspaces. - From Azure Monitor Workbooks.	- Start Logs from the menu for the Azure resource. - Start Logs from Azure Monitor menu. - Start Logs from Log Analytics workspaces. - From Azure Monitor Workbooks.

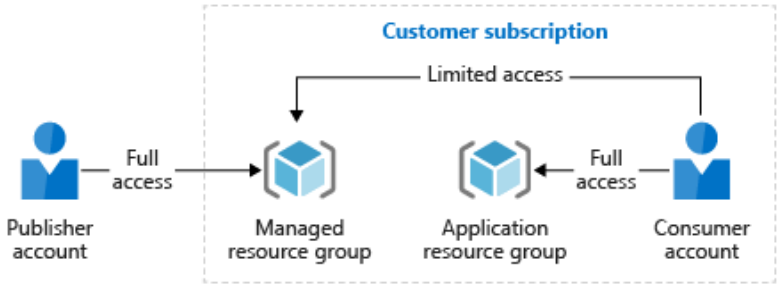
As you can imagine, the recommendation is to use workspace context for teams that need the full Azure Sentinel experience and resource context for application teams.

For a full discussion on this topic, visit this article.

Azure Managed Applications

There might be some situations where you want to keep your customer's Azure Sentinel environment locked down, so nobody from the customer organization can make changes to it. This would avoid situations where the customer accidentally deletes/disables analytics rules, removes playbooks or just close incidents. How can we achieve this?

In these situations, you can make use of Azure Managed Applications, which allow you to define infrastructure that will be deployed in a specific subscription but with the peculiarity that the resources can only be managed by the publisher, in other words: YOU. Not even a user with Owner role for the whole Management Group or Subscription will be able to modify the environment, they will only have **limited** access that you define.



Refer to the documentation for additional details on how to use this deployment method. The good news is that you can reuse existing ARM template samples to define your Azure Managed Application, for example, Azure Sentinel All-in-One.

Sizing & Pricing / Cost

Cost components

Azure Sentinel is billed based on the volume of data ingested for analysis in Azure Sentinel and stored in the Azure Monitor Log Analytics workspace. The analytics enabled by Azure Sentinel and do not include the related data ingestion charges for Log Analytics, and both costs must be considered within the estimation. Once Azure Sentinel is enabled on your Azure Monitor Log Analytics workspace, every GB of data ingested into the workspace can be retained at no charge for the first 90 days. Retention beyond 90 days will be charged per the standard Azure Monitor Log Analytics retention prices. Additional costs may be incurred based on usage of other services within Azure Sentinel. Utilizing Logic Apps for automation, or Machine Learning for analysis are two common examples.

Cost calculator for Azure Sentinel

Cost calculator for Azure Log Analytics

Azure Sentinel and Log Analytics offer ingestion & 90 day retention of some data at no cost. Those include:

- Azure Activity Logs
- Office 365 Audit Logs (all SharePoint, Exchange admin activity, Teams)
- Alerts from Microsoft Defender products (Azure Defender, Microsoft 365 Defender, Microsoft Defender for Office 365, Microsoft Defender for Identity, Microsoft Defender for Endpoint), Azure Security Center, Microsoft Cloud App Security and Azure Information Protection
- Azure Information Protection alerts (when available)
- Defender for IOT (alert)

Sizing and cost estimations

The key requirement in cost estimation is to identify the daily ingestion rate, usually in GB/day. In most cloud & hybrid environments, networking devices (firewalls, proxies, etc.), Windows and Linux servers produce most ingested data. An exhaustive inventory of data sources must be performed. The Azure Sentinel cost calculator includes tables useful to estimate footprints of data sources. These estimates are a starting point, but log verbosity settings and workload will produce variances. Regular monitoring is recommended based on your scenario. To understand ingestion patterns real-time, refer to the Log Data Usage and Costs help article.

Using this data, tailor the customer's capacity reservation, which can save up to 60% compared to pay-as-you-go pricing. Capacity reservations allow you to reserve a fixed amount of daily data ingestion capacity for Azure Monitor and Azure Sentinel for a fixed, predictable daily fee. You can upgrade your requested capacity at any time. However, the minimum commitment period before you can opt out or reduce your capacity reservation is 31 days.

- Adding more capacity to your reservation – You can upgrade your requested capacity at any time. Your new capacity reservation will be effective at the start of the next UTC day.
- Reducing your selected capacity reservation – You can reduce your capacity reservation or opt out entirely from the capacity reservation model after the first 31 days. This 31-day clock resets every time you make any change (increase or decrease) to your selected capacity reservation. Your new capacity reservation or business model choice will be effective at the start of the next UTC day.

For more details about capacity reservations a corresponding costs, visit the Azure Sentinel official pricing page.

Long term storage options summary

Customers normally need to keep data accessible for longer than three months. In some cases, this is just due to regulatory or audit requirements, but in some other cases they need to be able to run security investigations on data

that is older than 3 months. Whether to keep that data in Azure Sentinel or not depends on a number of factors, just to name a few:

- How often do you need to access data that is older than the retention period set in Azure Sentinel?
- What kind of access do you need? Is it just to show it to an auditor or you need to perform advanced hunting or investigations?
- How fast do you need the system to respond? Is it ok if the system takes a few hours to return the data, or you need it in a few seconds?

Here is a table that tries to summarize the long-term retention options for Azure Sentinel data:

Storage Options	Azure Sentinel	Azure Data Explorer	Azure Storage
Performance	High	High to Low (1)	Medium to Low
Maximum Retention	2 years	Unlimited	99 years
Cloud Model / Useability	SaaS / Great	PaaS / Good	IaaS / Fair
Cost	High	Medium	Low
Purpose	SecOps	Extended threat hunting, compliance, trend analysis, storage of non-security data, audit	Archive, Compliance, Auditing

(1) ADX performance varies depending on cluster size, SKU, and other factors

Let's now review each of them in detail.

Azure Log Analytics/Azure Sentinel

This option offers the best performance and provides all the advanced features available in Azure Sentinel, like ML based detection rules, visual entity-based investigation, incident management, UEBA, advanced threat hunting, etc. This is the best place for your security data, as you will benefit from all the security-focused functionality mentioned above.

- **Performance**: Azure Sentinel/ Azure Log Analytics offers high performance consistently, with low latency and automated data management. We don't have to worry about scalability and performance, as this is handled in the backend for us.

- **Usability**: Azure Sentinel is a full SIEM+SOAR solution and, as mentioned above, offers many features that make it essential in a SOC. **These features are unique to Azure Sentinel** and are not part of any of the other long-term storage options.

- **Cost**: The retention costs in Azure Sentinel are high compared to the other two storage options. Retention is charged per GB/month after the default 90-day retention has expired. Retention is applied to all sources, including the ones that are ingested for free.

Azure Data Explorer (ADX)

This option allows you store great amounts of data at a cheaper cost than Log Analytics/ Azure Sentinel, while still retaining some advantages. On one hand, you can use the same KQL queries in ADX as in Log Analytics/Azure Sentinel, so the same hunting/exploration queries can be used in both places. On the other hand, ADX offers new options to store the data and make it more performant and cost efficient. These options require user attention that is not needed in Azure Sentinel. As an example, in ADX you can define what data is available in hot cache and move specific tables to it on demand, so queries run faster. In the context of long-term storage, ADX is a great option if your team needs to run light investigations on the data stored in ADX, but without all the security intelligence built on top of Azure Sentinel. This option, and its different architecture alternatives, are explained in detail in [this blog post](#).

- **Performance**: several factors contribute to query performance of an ADX cluster: number of nodes in the cluster, cluster VM SKU, data partitioning. As you add more nodes to the cluster, the performance will increase, but also the price. You need to carefully size the cluster to find the sweet spot between performance and cost that works for your organization. This sweet spot will vary from one organization to another, highly dependent on how often the data is accessed and how quickly a response is expected.

- **Usability**: the usability of ADX in the context of security operations is good. You won't benefit from the many out of the box security features in Azure Sentinel (UEBA, visual investigation, incident management, etc.), but you can still explore the data using the same KQL queries you use in Azure Sentinel, and you can use visualization services like Azure Workbooks or Grafana on top of ADX data. Moreover, you can have an Azure Workbook that visualizes data spread across both Azure Sentinel and ADX. **You can also query ADX data right from within Azure Sentinel as explained here**, which also allows correlating data from both datastores. You can also do the opposite, query Azure Sentinel data from ADX (details here).

- **Cost**: As explained earlier, ADX can be tuned to offer the desired performance. It also offers autoscaling capabilities to adapt to workload on demand. On top of this, ADX can benefit from Reserved Instance pricing. You can run your own cost calculations in the ADX sizing tool. As the data needs to be moved from Azure Sentinel/Azure Log Analytics to ADX, there are several architectural options for this approach, and depending on which one you use, there might be additional costs to consider. For example, the Log Analytics data export feature has an associated cost that needs to be accounted for (info here). An additional cost component that you might need to consider is EventHub (used for queue management). EventHub is sized according to ingestion rate, so the faster data is sent into Azure Data Explorer, the higher the cost in this component.

Azure Blob Storage

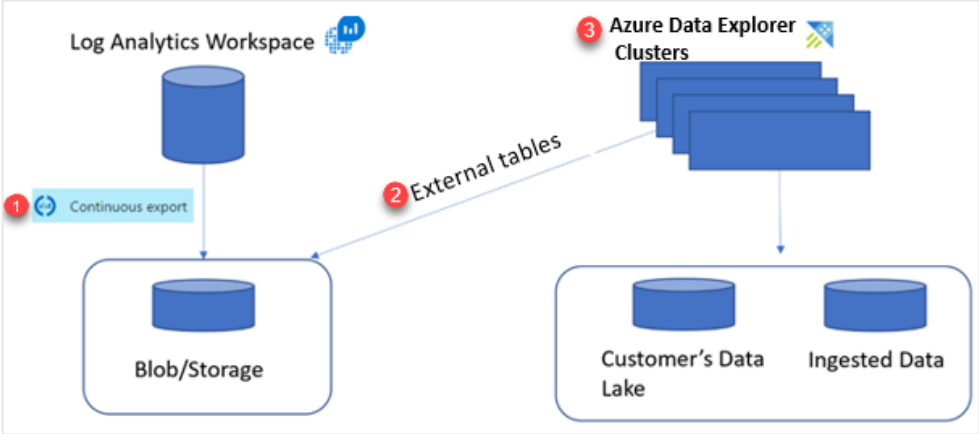
This option offers a very low-cost alternative that might be suitable for purely audit/compliance purposes. The data is still secure and accessible, but querying the data requires a bigger effort as the KQL queries need to be modified to include individual SAS URLs pointing to multiple blobs. This option is ideal for scenarios where we need to keep the data to comply with regulation standards and frequent data access is not expected. There are two alternatives to send log data to Azure Blob Storage: using a Logic App or using the Data Export feature in Log Analytics. In order to decide between these two options, take into account the Logic App connector limits, if you expect to go beyond these limits, you should choose the Data Export option.

- **Performance**: Azure Blob Storage offers two performance tiers: Premium or Standard. Although both tiers are an option for long term storage, Standard is generally chosen due to greater cost savings.

- **Usability**: usability is the biggest concern when choosing Azure Blob as your long-term storage option. When we use Azure Blob, the data is saved in separate containers (folders) for each data type and individual blobs in a folder for each hour in the day. This will result in a huge number of files after several weeks of data retention. The easiest way to explore this data is using the *externaldata* operator in KQL. You can see more details about the usage of this operator in this blog post.

- **Cost**: Azure Blob Storage is the cheapest storage option of all three, but you still need to account for the cost of the data export feature (if that option is chosen), which is charged by export GBs, or the periodic execution of the Logic App.

ADX and Blob Storage combined



This option allows you to have the best of both worlds. You send the data to Azure Blob's cheap storage, but you can still run KQL queries on the data almost as if you had it local in the ADX cluster. In [this article] you can see how to implement this approach. As you can see in the aforementioned article, this approach uses Log Analytics data export feature to send the data to Azure Blob Storage, but instead of querying the data from Log Analytics with the *externaldata* KQL operator pointing to each blob, you create an **external table** in ADX that points to the whole container. This way you don't have to reference each individual blob as this is managed transparently for you.

- **Performance**: Azure Blob Storage offers two performance tiers: Premium or Standard. Although both tiers are an option for long term storage, Standard is generally chosen due to greater cost savings.

- **Usability**: usability is the biggest concern when choosing Azure Blob as your long-term storage option. This is mainly because using *externaldata* operator makes it very challenging when you have a big number of blobs to reference. With this approach we **completely eliminate that burden using external tables in ADX**. The external table definition understands the folder structure in the blob storage account and allows us to query the data contained in many different blobs and folders transparently.

- **Cost**: In this architecture, ADX plays a very important role, but the good news is that the size of the cluster doesn't matter because ADX only acts as a proxy. As you can see in the architecture diagram, we use the Log Analytics data export feature, so you will need to factor this into the cost. See below for a detailed cost simulation.

In summary, there are several options available to keep your data accessible while reducing costs. After capturing what are your needs in terms of access to the data and performance you can use this guide to decide the appropriate platform to store your logs.

Data Collection

Use cases review

The quickest and easiest connectors to enable are the Direct ones that come with Azure Sentinel (see below for details of connector types). The table highlights some favorites:

Connector	Description	Why?	Free ingestion
Azure Active Directory	Includes the identity sign-in and audit logs (additional logs currently in preview)	Whenever a user attempts to log into any Azure service and if successful what they do.	No
Office 365	Includes user and admin activities from SharePoint, Exchange and Teams	Often a rich source of information and be used to run queries across to see what people have been up to.	Yes
Azure Activity	Records everything that happens within the subscription, which is where Azure Sentinel will exist.	Often quite small amounts of data but often very valuable for the who, what and when.	Yes
Microsoft 365 Defender	Takes an incidents / alerts from the family of Defender products	If the tenant has M365 Defender activated then provides a rich seam of security logs. Note, alerts only, raw data is not free!	Yes
Azure Security Center	Collect alerts from Azure Defender.	If any of the Azure Defenders have been activated (IoT, Kubernetes, SQL to name a few) then will ingest the alerts.	Yes

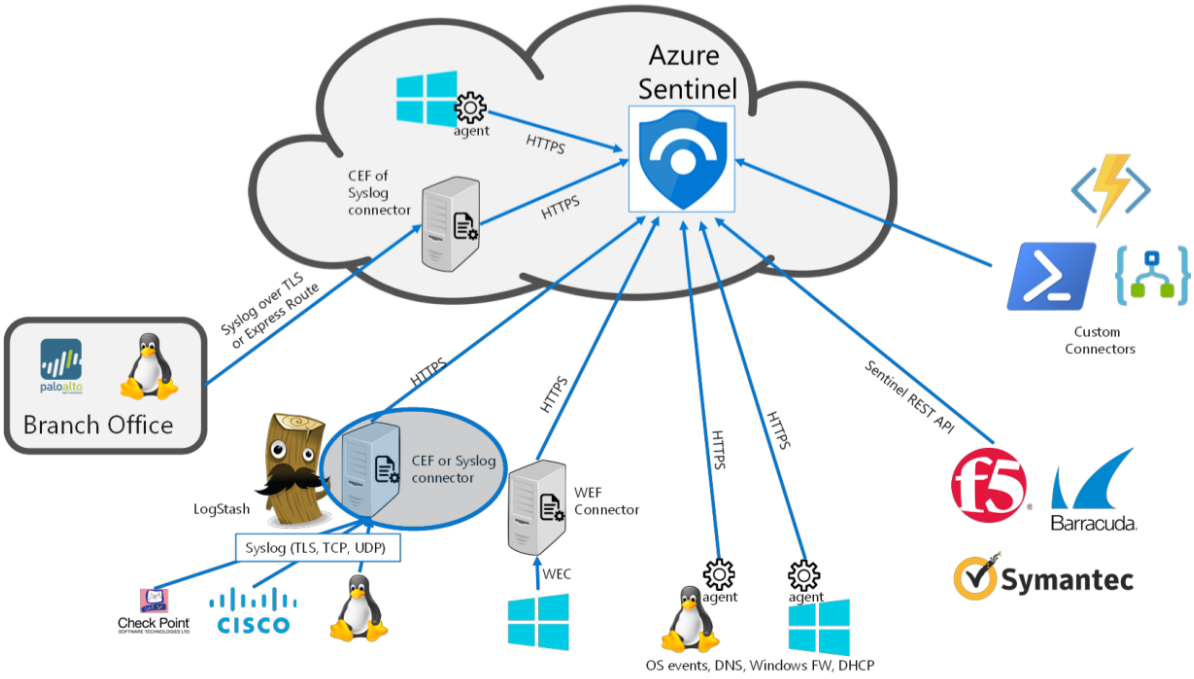
Data Sources collection overview

After the relatively simple task of created a workspace and enabling Azure Sentinel the real work begins with connecting data sources into Azure Sentinel. This is one of the areas where we see MSSPs adding value for their customer base with their skill and experience in connecting the correct log flows into Azure Sentinel without simply enabling all of them and the associated ingestion costs!

It is not an exaggeration to state that any security log of pretty much any description can be ingested into Azure Sentinel by using one of the connector types below. Even a CSV file can be ingested if really required, therefore supporting air gapped networks. To support this critical ingestion task there are five core categories of connectors that Azure Sentinel uses.

The five types of connectors are:

- **Direct** - native connection into Azure Sentinel, often with additional features beyond just log collection.
- **Syslog** - using the CEF format or even native Syslog, a collector server is needed for this connector type
- **Agent** - Both Linux and Windows devices can install a monitoring agent to collect logs to send into Azure Sentinel. Note, this agent is slowly being replaced by the Azure Monitor Agent.
- **Threat Intelligence** - used to ingest commercial IOCs from security partners.
- **Custom** - involve more work to create but ultimately allows for more control and scale as well.



Connector types

Direct connectors

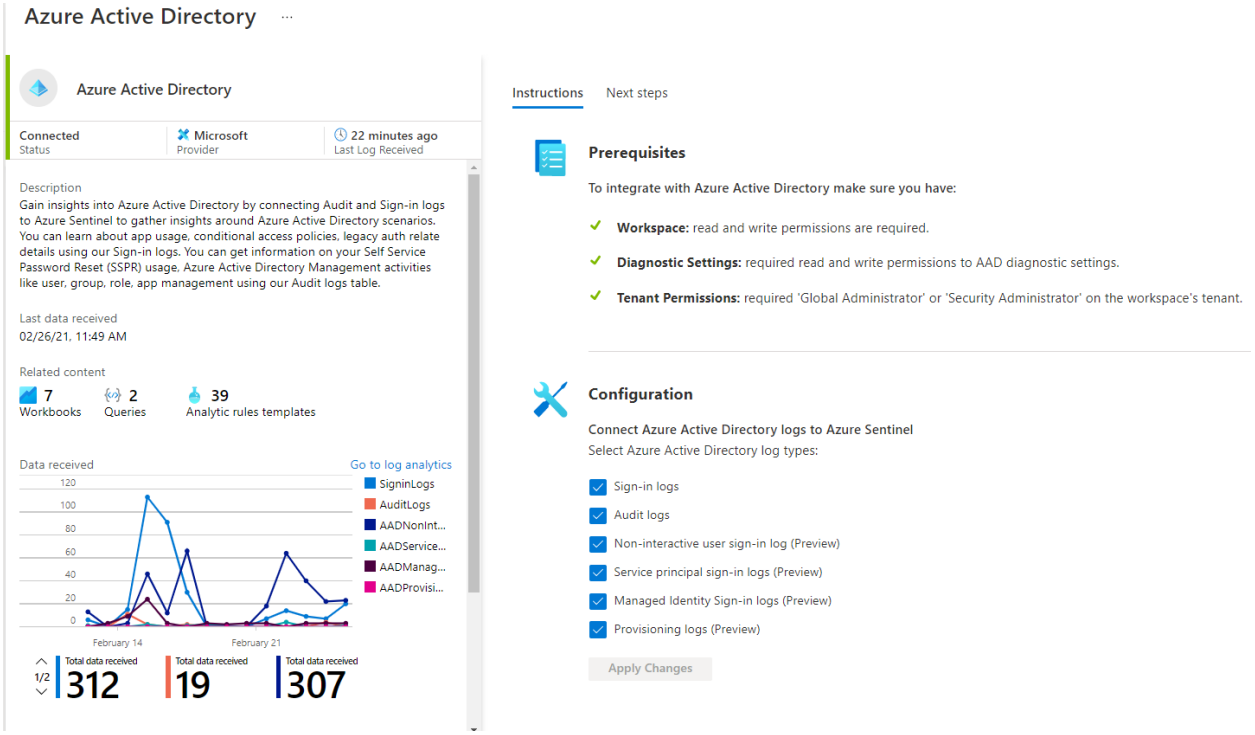
These connector types are available by default from Azure Sentinel, they are essentially built in. These data sources have native connectivity which implies a cloud type of service from Azure, AWS and in the near future GCP as well. However, today the list of direct connectors is heavily Azure flavored. Some common connectors used when Azure Sentinel is first built are:

- **Azure Active Directory** – Audit and Sign-in logs
- **Azure Activity** – logs subscription level events
- **Microsoft 365 Defender** – used to collect alerts from the Defender family. This will eventually include MCAS, Defender for Office 365, Defender for Endpoints and Defender for Identities. This is a feature rich connector which will also allow for raw data (as well as events or alerts) to be ingested if required and configured.
- **Office 365** - User and admin activities within Exchange, SharePoint and Teams. This connector uses the Office 365 Management Activity API, but ingests only alert level information NOT the raw data from this API.
- **Azure Security Center** - ingest alerts from Azure Defender

There is another category of direct connectors which reach out and connect directly to devices that can be located within the cloud but more typically on customer sites. These connectors use the source device API (when available) and are normally built by the device manufacturers. Some common direct API connectors are shown below:

- Barracuda WAF and Firewall
- Citrix Analytics
- F5 Big IP
- Forecepoint DLP
- Proofpoint
- Qualys VM (Azure Function based)
- Salesforce Service Cloud (Azure Function based)

The screenshot below shows the most commonly connected first security data source, Azure Active Directory sign-in and audit logs. Note the permissions needed to deploy the connector. Note, the license requirement is being retired imminently. You will also notice additional Azure AD sources currently in preview.



Syslog connectors (Forwarders)

Many traditional security products (and many others) will support sending security logs using syslog, a protocol used for sending security event information. Azure Sentinel does have the ability to ingest raw syslog messages (via a collector) but the preferred approach is to use CEF (Common Event Format) formatted events transported over the syslog protocol.

CEF data is normalized before being stored in Log Analytics and so is easier to query. Syslog data is not normalized and therefore more complicated to query. Though Azure Sentinel, unlike many other SIEM products, can still store and query syslog events.

Either way both types are supported via a collector. This collector uses a Linux machine as a log forwarder, to err forward events into Azure Sentinel. There is no direct connection from a syslog (or CEF) source to Azure Sentinel. For this to happen the device would need a well-documented API that allows security events to be read. See previous section on Direct connectors.

This collector can be installed on-premise or indeed in the cloud as a VM depending on the infrastructure available to the customer and MSSP. There are various guides on how to scale syslog collection, and at truly high ingestion volumes it may be more appropriate to use Azure Functions to allow ingestion to scale higher. Typically, daily ingestion amounts of over 1TB could need this investment.

There are potentially thousands of CEF and syslog connected devices, many are included within Azure Sentinel and include a link to a manufacturers page with help on configuring CEF / syslog. In this article we maintain a list of different data sources and the preferred collection method.

Agent based connector

There are currently two agent types to choose from. There is the 'classic' and its soon to be seen replacement. The classic one follows.

Azure Log Analytics agent (classic)

Also historically called the OMS agent or even MMA (Microsoft Monitoring Agent) which gives an idea of its long history. The agent runs on Windows and Linux machines.
This agent is typically used by Azure Sentinel to collect OS Events. It also collects logs from Microsoft DNS Servers, Windows Firewalls and Windows Security Events.
There are many other types of data that can be collected by this agent depending on the role of the machine the agent finds itself on. This even includes importing files. More details can be found here.

Azure Monitor Agent (new)

The incoming replacement for the Log Analytics agent is the Azure Monitor Agent. A public preview version of this agent already exists, though with limited functionality when used with Azure Sentinel. A more feature rich public preview is due by summer 2021.
This update agent is expected to support:

- An ability to filter events before ingesting into Azure Sentinel
- Support for multi homed solutions on Linux platforms
- Support for Windows event collection and filtering

Threat Intelligence connector

Note: There is a separate section focused on Threat Intelligence later on in this document.

There are two methods of connecting commercial Threat Intelligence Providers (TIPs), or in fact any source of threat intelligence into Azure Sentinel. Threat Intelligence can be any source of threat information though is often represented by a file hash, a URL or an IP address.

Microsoft Graph API

A rich interface that allows TIPs to feed a wealth of information into Azure Sentinel (and also Defender for Endpoint if needed!). More details can be found here. A number of commercial providers provide a suitable Graph API connector including:

- MISP
- Palo Alto MineMeld
- ThreatConnect
- ThreatQ

TAXII connector

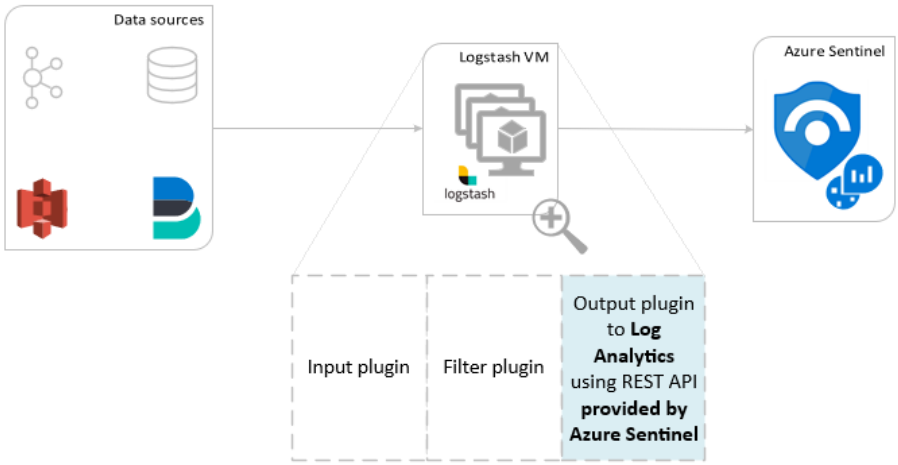
Azure Sentinel comes complete with a STIX/TAXII v2 (only, no v1 support) connector which enables a built-in TAXII client in Azure Sentinel to import threat intelligence from TAXII 2.x servers. . No need to create an Azure AD enterprise application just select the feeds required and any needed account information and the threat data will start to flow into Azure Sentinel.

As a final note it is possible to add custom (via the dashboard) indicators directly into Azure Sentinel as well.

Custom connectors

There are several methods available to connect data into Azure Sentinel if the above four categories cannot connect the source required.

Logstash
Instead of using the Log Analytics agent, Logstash can be used instead, and it gives some additional features over the default agent. There is an output plugin for Logstash that will connect it directly into Log Analytics where the logs are stored within a custom table. The diagram below shows the different elements of a Logstash solution.



The input plugin allows for customization of the collection of data from different sources.
The filter plugin allows for normalization of the data before being ingested into Azure Sentinel.
The output plugin is the connector provided by Azure Sentinel to connect the data.

Note, Logstash uses custom tables, not built-in tables that first party connectors for Azure Sentinel write to. This will be updated in the future, but for now certain Azure Sentinel features such as UEBA and Fusion can only reason over the built-in tables, not custom tables.

NOTE: the Logstash output plugin is currently in public preview.

More information about Logstash and its connection into Azure Sentinel can be found here.

PowerShell cmdlet
There is a cmdlet that can be found here which will allow upload of data into a custom table (see note above for restrictions with custom tables) within Log Analytics. This even includes flat files such as a CSV.

Log Analytics Data Collector API (public preview)
Like the cmdlet above, Log Analytics also has a Data Collection API which uses HTTP to stream data into Log Analytics and is in fact a standard REST API. This means any popular programming language can be used to send data into Log Analytics and therefore Azure Sentinel.

More information on this API can be found here.

Azure Logic Apps
Data collection can be orchestrated! A scheduled task or triggers can be used to connect to data sources and ingest this data into Azure Sentinel. Although this can be a relatively simple method for automating the ingestion of data is not recommended for larger data sets due to the underlying costs. Therefore, use for low volume data sources only or for enriching other data ingestions.
Some high-level examples of using Logic Apps to ingest data could be:

- Connecting to any REST based API
- Connecting to a SQL server
- Connecting to a file system

Note: Logic Apps can be used with on-premises equipment using gateways such as the On-Premises Data Gateway.

Azure Functions
As the name implies, a built-in Azure feature that can be used for Azure Sentinel purposes. It is a way of running serverless code and is generally used as an Azure Sentinel data connector when ingesting large amounts of data or data with unique collection requirements.

These types of connectors need to be built and a wide range of programming languages such as PowerShell, Node.JS, Python, .NET C# and Java are included.

Most of the newer connectors being built for Azure Sentinel use this collection method. You can see some examples in our official documentation, like Okta or Proofpoint TAP.

There are other connectors that are built by the community, like the Office 365 Management API connector, which collects not just the alerts that the standard built in connector handles, but additional audit logs. This can be found here.

The Azure Sentinel product team and the community often share new examples of Custom Data connectors. These are available on GitHub.

Connectors a final note

Getting data into Azure Sentinel is perhaps the key activity for an MSSP supporting a range of partners. Knowing what type of connector to use when to ingest quality security data is an area of value add.

Though, not all data is created equal. There is a concept of the processing pipeline for data which follows the four areas listed below:

Parsing
Azure Sentinel supports parsing at query time which means security data can be ingested in its original format. This means connectors do not necessarily need to be updated if the data source structure changes. Perhaps due to a firmware update on the source system, an operating system update, new product range etc. Instead, parsing is done at time of query or even during an active investigation.
However, the better formatted the original data source is, the easier it is to query. Hence, CEF is preferred over the raw Syslog data sources.

You can find KQL parsers for major vendors here.

Filtering
Not easily accomplished currently. The inbound Azure Monitor Agent will allow for some filtering otherwise a custom style connector (e.g., Logstash) is required to filter events before ingestion.

Normalization
Having a set format around the data ingested including fields used, data formats, how time is recorded, categories and resolution (usernames, device names, IP addresses etc.). Correlation between different data sets (from different data sources and therefore connectors) is a key skill when threat hunting or building analytic rules.

More information around Normalization for Parsing can be found here.

Enrichment
Generally, involves tying together data from different sources to provide a richer stream of data, and to enhance incidents when they occur. This is typified by adding areas such as GeoIP information, data form WhoIs, username and device name enrichment as well.

A common example is enriching suspicious file information from VirusTotal which can be automated using Logic Apps for example. More details on this approach here.

Connector links

- Connectors – The Grand List - here
- Create a custom connector - here
- Log Analytics Agent – some updates - here
- Azure Monitor Agent – preview - here
- Data sources (another collector list) - here

Ingesting Microsoft 365 Defender Advanced Hunting logs

As mentioned at the beginning of this section, there is a new data connector that allows customers to stream M365 Defender raw data into Azure Sentinel. This option enables additional threat hunting, correlation, applied threat intelligence, and advanced machine learning algorithms.

The data can also be sent to the security data warehouse (ADX) for long term data retentions. This could occur at the same time as sending to the SIEM, or it can be sent there after the SIEM has expired the data.

Automation/SOAR

Built on the foundation of Azure Logic Apps, Azure Sentinel's automation and orchestration solution provides a highly extensible architecture that enables scalable automation as modern technologies and threats emerge. You can use these features to automate your common tasks and simplify security orchestration with playbooks and automation rules (Preview) that integrate with Azure services as well as with your existing tools.

Automation Rules

What is it?
Automation rules allow you to centrally manage all the automation when it comes to incident handling.
Automation rules streamline automation use in Azure Sentinel and enable you to simplify complex workflows for your incident orchestration processes.

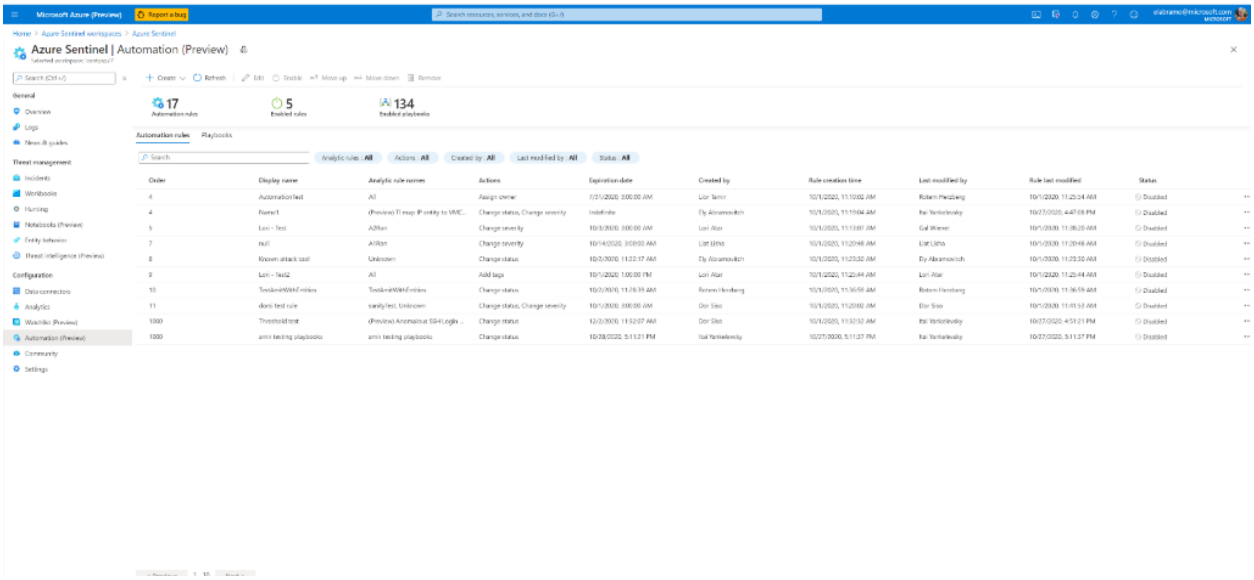
How does it work?
Automation rules are triggered by the creation of incidents. You can set conditions to govern when actions will run, based on the incident properties, entity details and/or analytics rules that triggered them. You can also set the **order** of actions and the rule's expiration time.

Automated incident response
Explicitly set incident status or severity, assign an owner, or add a tag when an incident is created, **without** the need to run a playbook**.**

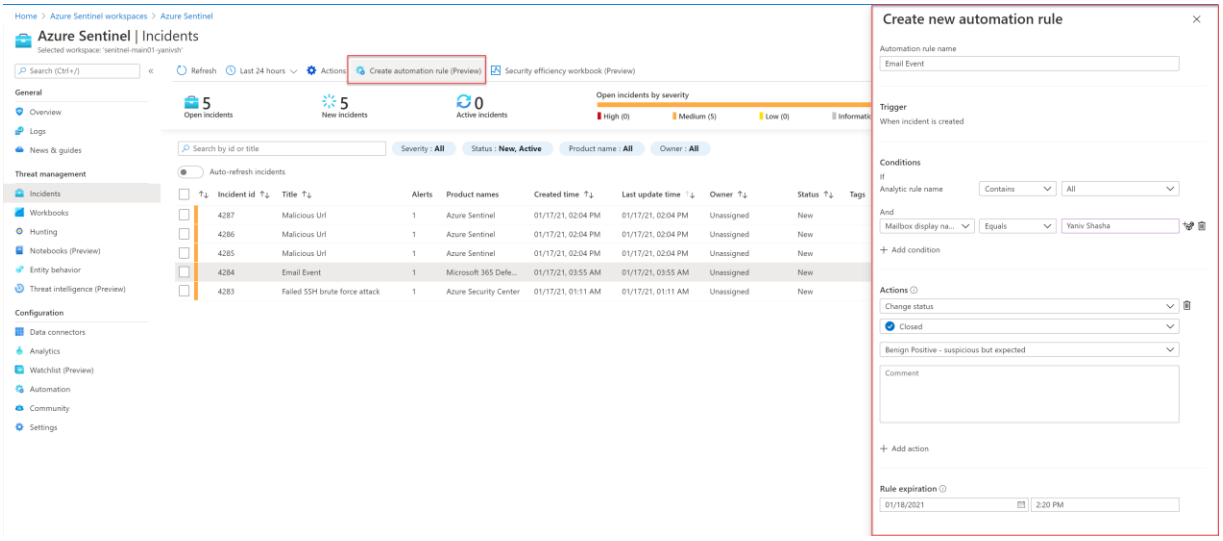
1. **Running playbooks on incidents**
   You can still run playbooks from your automation rules to integrate with other services or create complex workflows. These automation rules, pass all incidents details to the playbooks, including alerts and entities.
2. **Trigger playbooks for Microsoft Providers**
   Automate the handling of Microsoft security alerts by applying automation rules to incidents.
   created from alerts.
3. **Do more with your playbooks**
   Automation rules allow you to Apply a single playbook to any of your analytics rules once, attach multiple playbooks to a single automation rule, and control the order of playbook execution. This will allow you to create simpler playbooks, easier to maintain and test, and arrange them in combinations as needed.
4. **Apply incident suppression**
   You can use rules to automatically resolve incidents that are known as false positives. For example, when running penetration tests, during scheduled maintenance or upgrades, or testing automation procedures.

Creating and managing automation rules

Automation rules are centrally managed in the new Automation tab under the Automation Rules sub tab (see screenshot below). From there, users can create new automation rules and edit the existing ones. They can also drag & drop automation rules to change the order of execution and enable or disable them.



Automation rules can also be created from within the Incidents view. Here is a screenshot that shows you how to create an automation rule from a given incident:



Azure Sentinel Playbooks

A security playbook is a collection of procedures that can be run from Azure Sentinel in response to an alert. A security playbook can help automate and orchestrate your response and can be run manually or set to run automatically when specific alerts are triggered. Security playbooks in Azure Sentinel are based on Azure Logic Apps.

Azure Sentinel and Logic Apps integration

A security playbook is a collection of procedures that can be run from Azure Sentinel in response to an alert. A security playbook can help automate and orchestrate your response and can be run manually or set to run automatically when specific alerts are triggered.

Security playbooks can be run either manually or automatically. Running them manually means that when you get an alert, you can choose to run a playbook on-demand as a response to the selected alert. Running them automatically means that while authoring the correlation rule, you set it to automatically run one or more playbooks when the alert is triggered. Here are the two types of triggers that we support:

Alert (Manual) Trigger
- When triaging an incident
- Invoking from the Investigation view

Incident Trigger for Automated Response
- Scheduled Analytics Rule
- Connector based Analytics Rule (aka first party alerts)
- Multiple Playbooks trigger for Incidents

The product team and the community shared many examples for LogicApp templates that available on Github

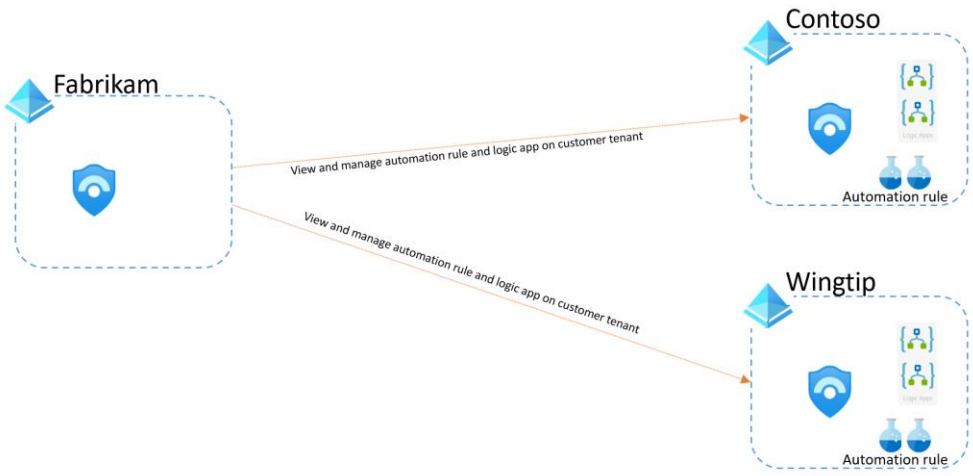
MSSPs design considerations for automation rule and playbooks

As explained above, when dealing with SOAR in Azure Sentinel, we have two main components.

- Automation rule
- Playbook (Azure logic App)

Depending on the MSSP needs, we have two main models on how to deploy both playbooks and automation rules:

Model 1: Automation rules and Logic app components stored on the customer tenant.

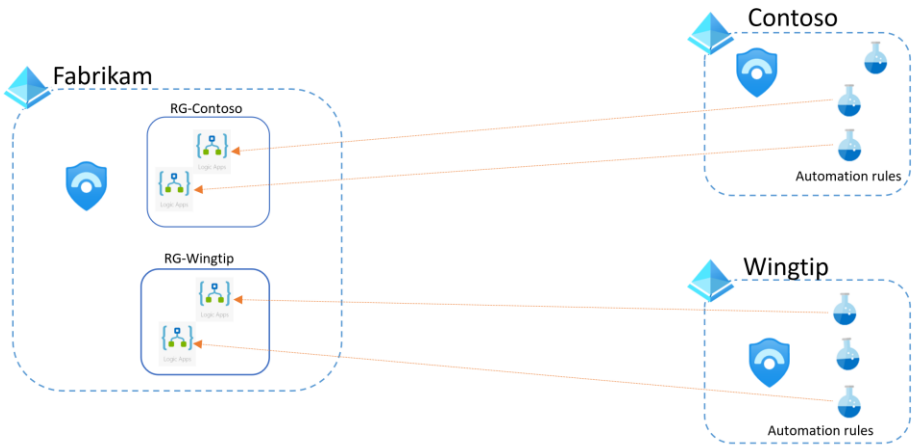


In this above example, the partner manages two customers over Azure lighthouse, the SOAR artifacts like the automation rules and playbooks are stored and run on the customer's tenant.

Considerations for this model:

- Logic app execution price is charged to the customer
- Monitoring for success or failure needs to happen against the customer environment

Model 2: Automation rule run on customer tenant and logic app stored and run on the partner side



In this setup, the automation rule that runs on the customer side calls a Playbook on the MSSP side. This approach is **recommended when the MSSP needs to protect the Intellectual Property** built into the playbooks.

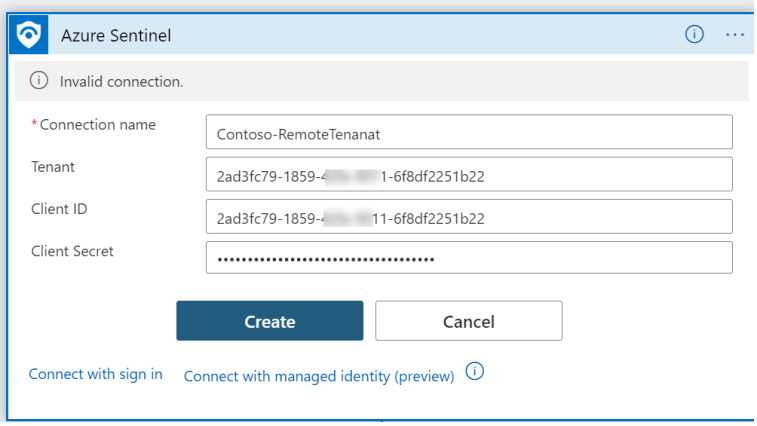
Considerations for this model:

- Logic app execution price is charged to the MSSP.
- Recommendation is to place the playbooks in a separate resource group per customer.

In model number 2, if the playbook needs to perform any actions in the customer tenant, we will need to create an identity object (SPN) **in that customer's tenant**.

After creating the SPN, we need to **make sure it has all the appropriate permissions** to perform all the actions required by the playbook. For example, if the playbook needs to block a user in Azure AD, the SPN will need to have the corresponding permissions that allow this task.

In the MSSP tenant, configure the logic app actions to use the identity object (SPN) that we created in the customer tenant:



You will need to update other connectors to use the SPN as needed.

In summary, the SOAR capabilities in Azure Sentinel can be a huge help for MSSPs, greatly reducing the amount of time spent triaging and investigating incidents.

Threat Intelligence

What is it?

Cyber threat intelligence is information describing known existing or potential threats to systems and users. This type of information takes many forms, from written reports detailing a particular threat actor's motivations, infrastructure, and techniques, to specific observations of IP addresses, domains, and file hashes associated with cyber threats. Threat intelligence is a primary data set used by SOC analysts to aid in detection of potential threats, prioritizing incidents, and contextualizing malicious activity. Our goal with Azure Sentinel is to ensure that you can easily incorporate and effectively use threat intelligence to enhance and enrich all areas of the SIEM.

Threat Intelligence can be sourced from many places, such as open-source data feeds, threat intelligence-sharing communities, commercial intelligence feeds, and local intelligence gathered in security investigations within an organization. Azure Sentinel lets you import the threat indicators your organization is using, which can enhance your security analysts' ability to detect and prioritize known threats.

How do you stream threat indicators to Azure Sentinel?

You can stream threat indicators to Azure Sentinel via the following built-in connectors and custom methods:

1. Using one of the integrated threat intelligence platform (TIP) products such as MISP Open Source Threat Intelligence Platform, Anomali Threat Stream, Palo Alto Networks MineMeld, ThreatConnect Platform, ElecticIQ Platform, ThreatQ Threat Intelligence Platform.
2. Connecting to TAXII servers
3. Using direct integration with the Microsoft Graph Security tiIndicators API
4. Importing custom threat indicators via Threat Intelligence blade on the Azure Sentinel portal

Our recommendation is to have the same set of threat indicators imported to your customers' Azure Sentinel workspaces. Then in your MSSP workspace, you can leverage cross-workspace queries to aggregate the threat indicators from the customer's workspaces and correlate them with your incident detection, investigation, and hunting experience.

The imported threat indicators can then be viewed and managed from the **Threat Intelligence blade** and queried from the **ThreatIntelligenceIndicators** table in the Logs. Threat indicators will not be deleted from Log Analytics even when they expire. To access only your active threat indicators, add the following filter clause in your queries:

```
| where Active == true
```

Several features from Azure Sentinel then become available or are enhanced and should be utilized:

- **Analytics** includes a set of out-of-the-box scheduled rule templates you can enable to generate alerts and incidents based on matches of log events from your threat indicators. A typical rule using threat data starts with "TI map" in its name.
- **Threat Intelligence workbook** provides summarized information about the threat indicators imported into Azure Sentinel and any alerts generated from analytics rules that match your threat indicators.
- **Hunting** queries allow security investigators to use threat indicators within the context of common hunting scenarios.

- **Notebooks** can use threat indicators when you investigate anomalies and hunt for malicious behaviors.
- **Playbooks** can be used to enrich alerts and remediate threats with threat data. Azure Sentinel GitHub repo has a collection of playbook templates built by both Microsoft and our partners (i.e., RiskIQ, Recorded Future, HYAS, etc.)

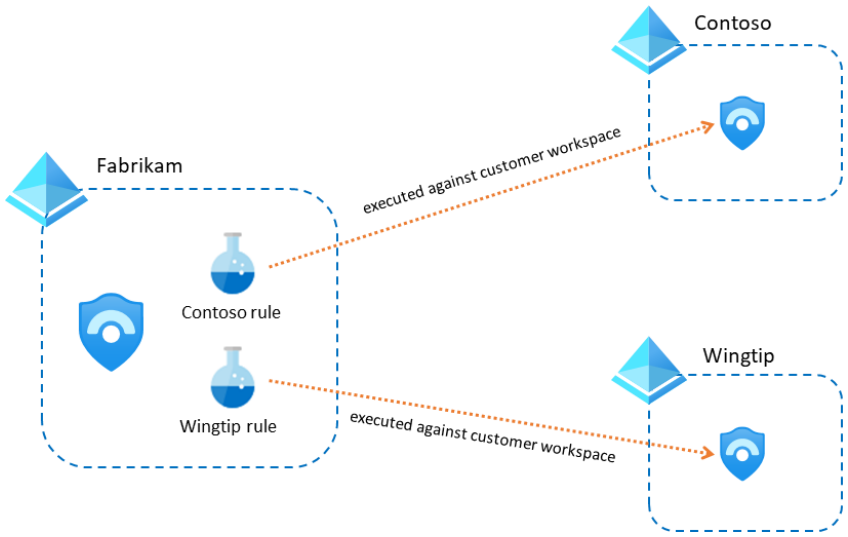
Analytics Rules

Once you have connected your data sources to Azure Sentinel, you'll need to create queries that will execute on a schedule and detect suspicious activities. Azure Sentinel comes with more than 180 built-in templates written by Microsoft's security experts and analysts based on known threats, common attack vectors, and suspicious activity escalation chains. You have more information about analytics rules in general here, and specific to scheduled rules here.

But what is specific to MSSPs when it comes to Analytics Rules?

Cross-workspace Analytics Rules

As with other resource types, the key feature is to make Analytics Rules work across workspaces and ultimately across Azure AD tenants, these are called cross-workspace Analytics Rules. Here is an example:



In this setup, we call ***local*** the workspace created in the MSSP tenant and ***remote*** the one created in the customer tenant.

The KQL query of *Contoso rule* would be like this:

```
workspace('contoso_workspace').SecurityEvent
| where EventID == '4625'
```

If required, we can also add multiple workspaces (up to **20** workspaces) to the query, so the rule can aggregate or correlate data from multiple workspaces:

```
workspace('contoso_workspace').SecurityEvent
| union workspace('wingtip_workspace').SecurityEvent
| where EventID == '4625'
```

As you can imagine, if you add 20 workspaces to the query, it can become easily unmanageable. In order to make these queries easier to use and read, you can create a **KQL function** that serves as an **alias** for a query that contains multiple customer workspaces. For example, saving this query:

```
workspace('contoso').SecurityEvent
| union workspace('wingtip').SecurityEvent
```

as a function named *ContosoWingtip_SecEvents,* that way, you can write you analytics rules and hunting queries like this:

```
ContosoWingtip_SecEvents
| where EventID == '4625'
```

Also, take into account that KQL functions can be automated via PowerShell, ARM or API, so you could easily setup automation to update your KQL functions whenever you add a new customer.

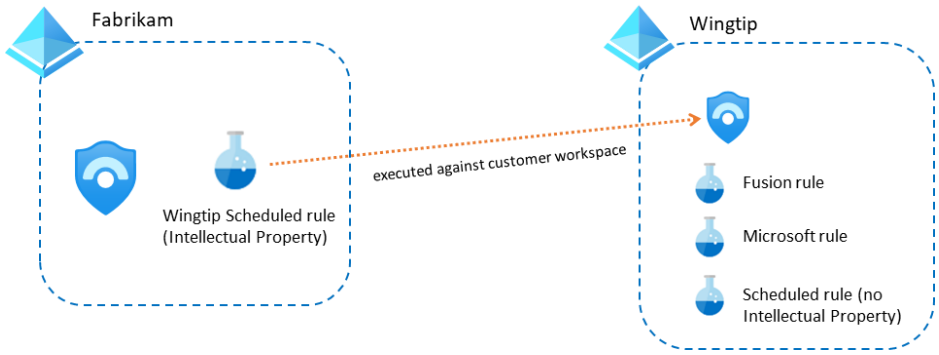
Some important facts about cross-workspace analytics rules:

- The **Azure Sentinel solution needs to be installed on both source and target Log Analytics workspaces**.
- You can only add **up to 20 workspaces** to a given cross-workspace analytics rule, although we recommend keeping it under 5 for good performance
- Azure Sentinel **incidents and alerts** raised by a cross-workspace analytics rule, **will only be created in the workspace where the rule was defined** (they will not show up in the "remote" workspaces)
- Entities from the remote tenant would be available in the source workspace, but visual investigation won't fully work
- They are **only available for Scheduled analytics rules**. Fusion, Microsoft, and ML Behavior Analytics will only look at the data in the workspace where these rules are enabled
- **Use caution**. As you add more workspaces to the query, the performance may decrease, and rules might end up failing. If possible, avoid adding multiple workspaces into a rule if there's not a strong reason to do so. Further information about query optimization can be found here.

When should we use cross-workspace analytics rules?

- When the analytics rule needs to correlate data stored in multiple workspaces (rarely needed in MSSP setup)
- To protect the Intellectual Property created as part of an analytics rule

Apart from these two scenarios, the scheduled analytics rule should be created in the customer workspace. Here is a sample on how this could look like:



As you can see above, we only create in our tenant the analytics rules that contain MSSP intellectual property. The rest should be created in the customer workspace. Also notice that in this situation you will have alerts and incidents in both workspaces (local and remote).

Managing analytics rules in the MSSP tenant

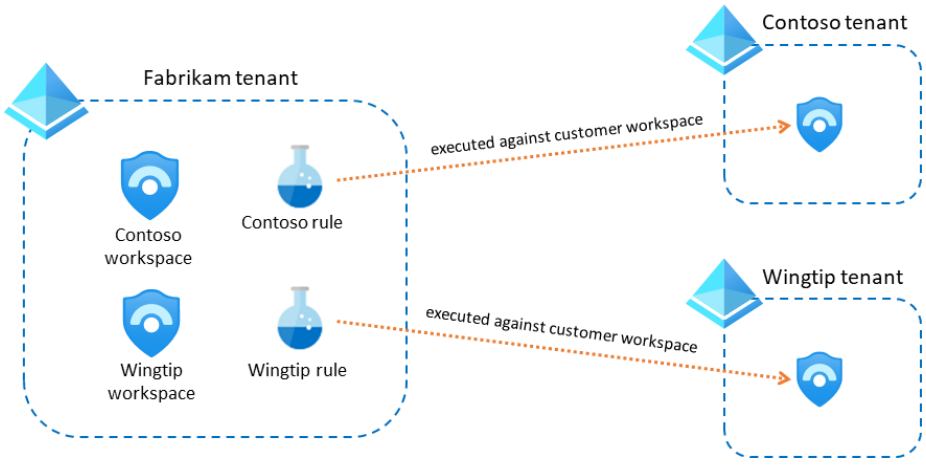
Just as a reminder, you should only **create rules in the MSSP tenant if you need to protect your intellectual property**. If that's the case, you would need to manage those rules appropriately.

As you may have noticed, in the examples above we always create a new analytics rule for each customer. For example, if we have a rule named "Suspicious logins in Azure AD", we will create one rule for Contoso named "Contoso – Suspicious logins in Azure AD" and another one for Wingtip name "Wingtip – Suspicious logins in Azure AD". Why not create a single rule that looks at multiple customer workspaces in parallel? There are several reasons:

- Query performance can be greatly impacted if customer workspaces are located in multiple regions.
- If an alert is triggered, it might be difficult to identity which customer is the alert coming from (there are methods to work around this, but they add complexity).
- The limit of workspaces in the alert rule is 20, so if you have more than 20 customers, you will still need to breakdown by groups of customers.

Because of the above reasons, we recommend having a single rule per customer as in the pictures above. As you can imagine, this can result in a high number of alert rules created in the MSSP tenant…imagine 50 rules per customer and having 100 customers, it would result in 5000 rules.

If you expect having a big number of proprietary rules and customers, you need to keep in mind that **an Azure Sentinel workspace has a limit of 512 analytics rules**. To work around this limitation, you could use an architecture where you create one Azure Sentinel workspace in the MSSP tenant for each customer that you manage. This would look like this:



In this architecture, **we also recommend hosting each customer workspace in a separate resource group**. This resource group can also be the container of other customer-related artifacts like playbooks and workbooks (see Automation chapter for similar architecture pattern).

Cross-workspace incident view

As explained in this article, you can see and manage incidents being raised across workspaces. This feature is documented here and currently **has a limit of 10 workspaces**. This limit is expected to increase in the upcoming months, but we need to keep in mind how many customers can a group of analysts monitor in parallel without being

overloaded and/or inefficient. If we assign a big number of customers to a team of analysts, it might not be able to cope with the number of incidents and the quality of the managed service offered to the customer will decrease.

As you may have noticed by now, **you may have alerts and incidents showing up in two or more workspaces for a given customer**. This can lead to situations where you can only monitor 5 customers (in the best-case scenario) with the current limit of workspaces in the incident view, because you have to select the local (where alerts/incidents coming from Fusion, ML Behavior Analytics, Microsoft and non-proprietary Scheduled rules are) and remote (where alerts/incidents coming from your Scheduled proprietary rules are) workspaces for each customer.

To alleviate this situation, you can use a workbook that shows full list of incidents regardless of the 10 workspaces limitation in the cross-workspace incident view. An example of this workbooks is the Azure Sentinel Central workbook shown here (see more in next section about Workbooks)

Rules Migration

When migrating from existing SIEM, the migration of Detection Rules usually becomes a requirement as part of the migration effort.

Please refer to the following content to learn mor**e about our recommendations for ways to migrate analytic rules from Splunk/QRadar/ArcSight:**
Webinar: Best practices converting detections rules from existing SIEM to Azure Sentinel
How do you export QRadar offenses to Azure Sentinel
Splunk to Kusto Query Language map

Azure Sentinel Workbooks

What is it?

Azure Workbooks provide a flexible canvas for data analysis and the creation of rich visual reports within the Azure portal. They allow you to tap into multiple data sources from across Azure and combine them into unified interactive experiences.

How does it work?

Workbooks can query data from multiple sources within Azure. Authors of workbooks can transform this data to provide insights into the availability, performance, usage, and overall health of the underlying components. For instance, security logs from Azure Active Directory and Azure Security Center can display the results as a grid in an interactive report allowing you to easily combine multiple data sources.

What does it do for you?

The real power of workbooks is the ability to combine data from disparate sources within a single report. This allows for the creation of composite resource views or joins across resources enabling richer data and insights that would otherwise be impossible.

Workbooks are currently compatible with the following data sources:

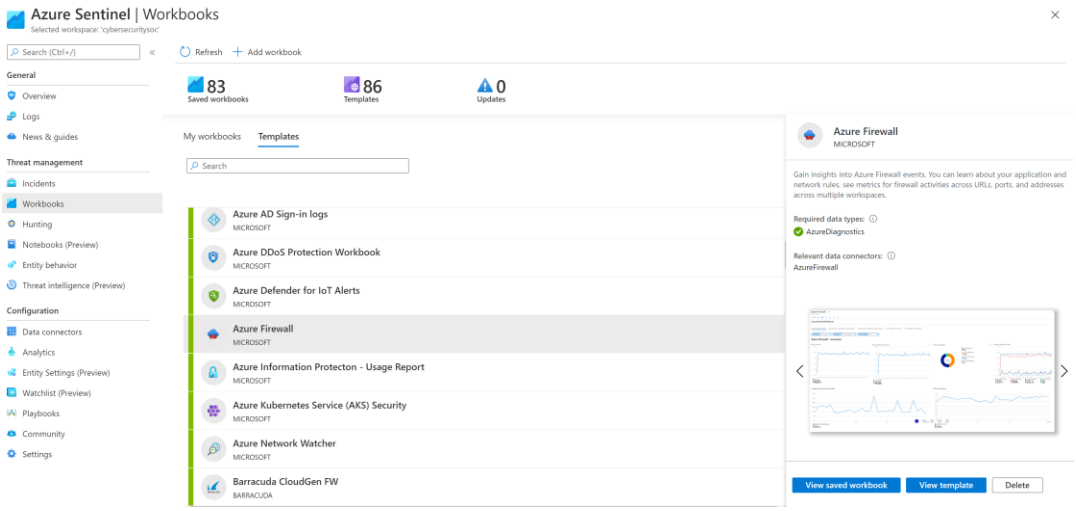
- Logs
- Metrics
- Azure Resource Graph
- Alerts (Preview)
- Workload Health
- Azure Resource Health
- Azure Data Explorer

Workbooks provide a rich set of capabilities for visualizing your data. For detailed examples of each visualization type you can consult the example links below:

- Text
- Charts
- Grids
- Tiles
- Trees
- Graphs
- Composite bar

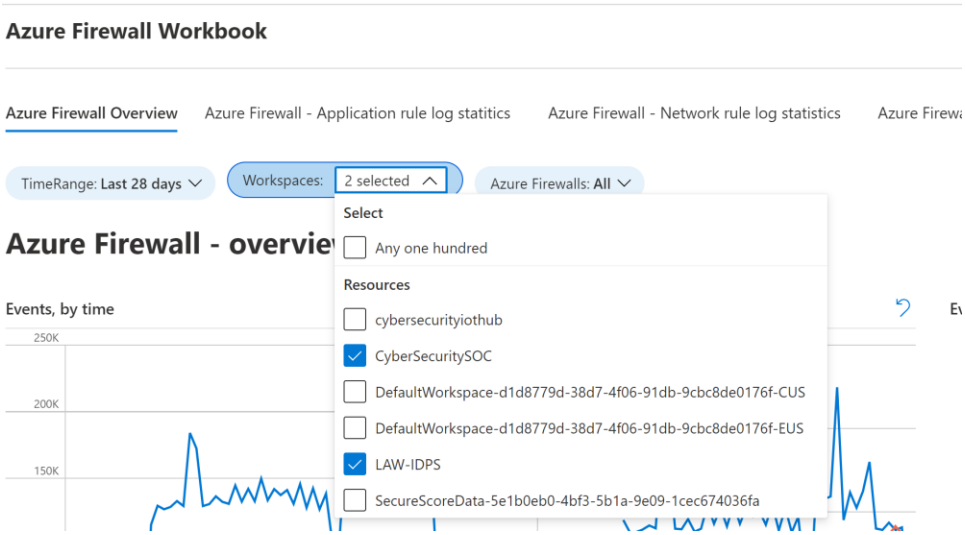
Azure Sentinel provides workbooks templates around each data connector, these templates can be used as references to add your own modifications or you can enjoy the updates of the templates provided by the Azure Sentinel community and the Azure Sentinel development teams.

Example of template management through Azure Sentinel:



Workbooks have a top-down approach to data manipulation, for example I'd suggest using the Azure Firewall Workbook template. This Workbook offers Time range, Workspaces (allowing cross-workspace queries) and specific resource filtering. You can add this to any Workbook to increase the range of queries, currently there are limitations on log analytics workspace queries which are listed here: Query across resources with Azure Monitor - Azure Monitor | Microsoft Docs (limit is currently 100 workspaces in a single query)

Example of a workbook with the multi-workspace filter:

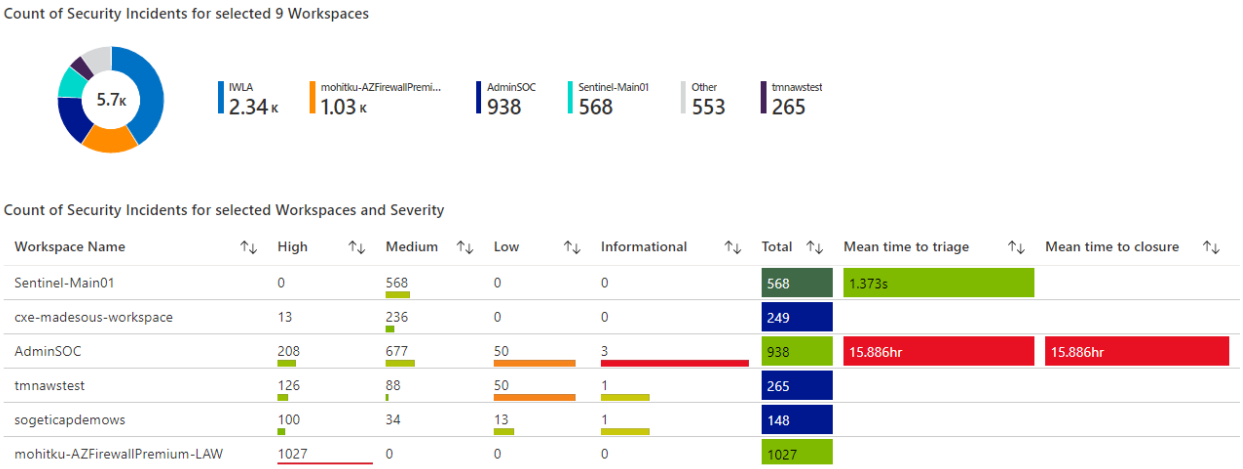


Azure Sentinel Central Workbook

There is one specific workbook that can be particularly important for MSSPs, its name is Azure Sentinel Central and is authored by Clive Watson.

This workbook provides you with a cross-tenant view of the different subscriptions and workspaces managed via Lighthouse. At the top, it includes filters so you can focus on specific subscriptions or workspaces.

In the first table, it will give you a view with the number of incidents (grouped by severity) per workspace and a couple of very useful statistics to measure your service performance: mean time to triage and mean time to closure.



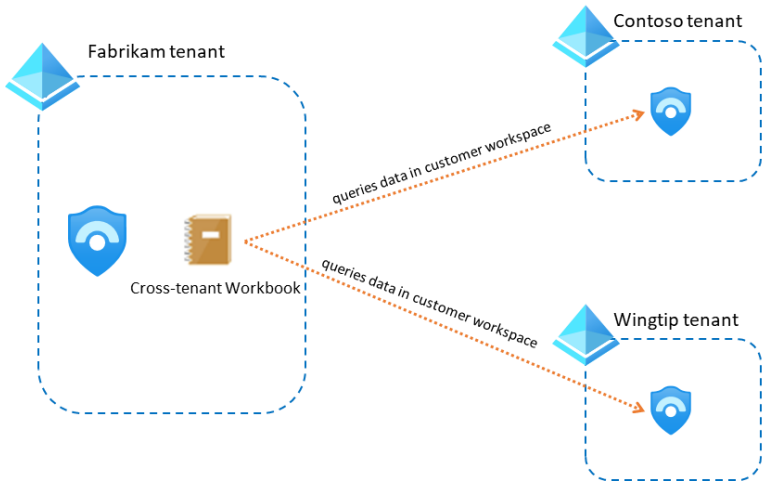
Below in the workbook you have a very useful view that has the list of incidents per workspace in the last 24 hrs, and links to jump to each specific incident. You also have the option to open the Investigation Insights workbook.

At the bottom you can see the full list of incidents for a given workspace.

Intellectual property protection

If you have developed your own intellectual property into a Workbook, you might need to hide it from your customers.

In order to do that, you can host the workbook in the MSSP tenant and make it multi-tenant as shown in the picture below:



This article explains how to do this step-by-step: [Making your Azure Sentinel Workbooks multi-tenant (or multi-workspace) - Microsoft Tech Community]

In this previous scenario, the MSSP will have cross-customer visibility, but the **customer won't be able to access the workbook**. What if the customer needs to see the workbook visualizations but we want to keep the code underneath secret? In this case, the recommended approach is to export the workbook to **PowerBI**, as explained here. This provides several **additional benefits**, to name a few:

- Easier to share. You can just send a link to the PowerBI dashboard and the user will be able to see the report. No need to have Azure access permissions.

- Scheduling. You can configure a PowerBI to send an email on a given schedule, that will contain a snapshot of the dashboard.

Other interesting workbook resources:

- Azure Sentinel Central workbook by Clive Watson
- Azure Monitoring Workbooks Video - https://www.microsoft.com/en-us/videoplayer/embed/RE4B4Ap
- Azure Sentinel Workbooks 101 - Azure Sentinel Workbooks 101 (with sample Workbook) - Microsoft Tech Community
- Azure Sentinel Cross-Workspace - Extend Azure Sentinel across workspaces and tenants | Microsoft Docs

DevOps - CI/CD automation

Automation and DevOps practices are crucial components for a successful managed security practice. These are some of the key benefits of good automation:

- Reduction of human error.
- Much faster deployment and configurations.
- Improved change management as changes are tracked in source code control.
- Enhance security as consistency is guaranteed.
- Time savings to allow employees to focus on adding value to our customers.

In this section we will try to review all the options and best practices to build a successful automation framework on top of Azure Sentinel.

Automation options

There are mainly 3 different mechanisms to automate your Azure Sentinel deployment: API, PowerShell, and ARM templates. Let's review each of them.

API

Azure Sentinel has its own RESTful API, which is called **SecurityInsights** (which is the name of the Azure Sentinel resource provider). This API is documented here and covers the following components:

- **Data Connectors** – CRUD operations
- **Alert Rules** – CRUD operations
- **Alert Rule Templates** – Get details about existing templates
- **Incidents** – CRUD operations
- **Bookmarks** – CRUD operations
- **Incident Comments** – Create and read operations
- **Actions** – CRUD operations for automation playbooks attached to rules

NOTE - As you may have noticed in the API documentation, the API version is 2020-01-01, which is the current stable API version. There is also another version (2019-01-01-preview) that is documented here but currently in preview and therefore subject to change. Be careful when using the preview version, as operations and schemas might change.

It's also important to note, that there are other Azure Sentinel components that are not covered by the **SecurityInsights** API. This is because they belong to other Azure resource providers as specified here:

- Hunting queries (Microsoft.OperationalInsights)
- Watchlists (Microsoft.SecurityInsights but not yet in the stable API)
- KQL functions (Microsoft.OperationalInsights)
- Workbooks (Microsoft.Insights)
- Playbooks (Microsoft.Logic)

All these APIs use the same authentication method as all other Azure Resource Manager APIs (see here for details).

Review this article if you want to get started with the Azure Sentinel API.

PowerShell

There are currently two PowerShell modules that can be used to manage Azure Sentinel components:

PS Module	Built-by	Support	API coverage	Technology	Documentation
Az.SecurityInsights	Microsoft	Microsoft official support	SecurityInsights 2020-01-01	Azure SDK for .NET	Blog and samples
AzSentinel	Community	Community support	SecurityInsights 2020-01-01, 2019-01-01-preview OperationalInsights (for HuntingRules and some connectors)	Wrapper around Azure Sentinel and ARM APIs	Repository

ARM templates

Most Azure Sentinel components can also be deployed and updated using ARM templates. You can find some samples here.

Update capability was recently added for Analytics Rules and Data Connectors, so you can now enable and update analytics rules and data connectors using ARM templates.

Automating Deployment and Configuration Management (CI/CD)

Imperative vs Declarative

There are fundamentally two programming paradigms that can be used in our DevOps implementation: Imperative and Declarative.

- **Declarative**: describes the end state that we want our environment to be like, without describing the steps that are needed to get to that state. In this approach, the computing platform (in our case Azure Resource Manager) decides which are the steps that need to be executed to my environment, so it matches the definition I provided. Examples of declarative programming are ARM templates and Terraform.

- **Imperative**: Provides the computing platform a set of instructions that need to be executed to get to the desired end state. Example of imperative programming are PowerShell and Ansible.

We will not go into the depths of which one is better over the other, each one of them has advantages and disadvantages.

Road to DevOps

In general, you will need to follow these four steps to have an end-to-end automation using DevOps practices:

1. Codify how your Azure Sentinel environment should look like.

2. Write automation that takes that code, interprets it, and deploys it to Azure Sentinel.

3. Create Continuous Integration pipeline/s that validate your source codes and warn you in case any mistakes are found.

4. Create Continuous Delivery pipeline/s that will kick off the automation written in step 2 and deploy/update resources in your Azure Sentinel environments.

There are a couple of prototypes built by the community that can be used to get started on this journey:

- Azure Sentinel as Code
- Azure Sentinel DevOps

Keep in mind that these prototypes are **community efforts** and as such, **not officially supported by Microsoft**.

Automating Azure Sentinel components

As we reviewed a little bit above, there are several ways to automate Azure Sentinel components. In the following table we try to summarize the options available for each Azure Sentinel component:

Component	API	PowerShell	ARM	Terraform
Onboarding	✓	✓	✓	✓
Connectors	✓	✓	✓	✗
Analytics Rules	✓	✓	✓	✗
Hunting Queries	✓	✓	✓	✗
Workbooks	✗	✗	✓	✗
Playbooks	✓	✓	✓	✓
Watchlists	✓	✗	✗	✗
KQL functions	✓	✓	✓	✓

We will now go a little bit deeper into each component:

Onboarding

The most common ways to enable Azure Sentinel on a workspace are PowerShell and ARM.

- *PowerShell*: New-AzMonitorLogAnalyticsSolution cmdlet from the Az module is the recommended approach. Sample available here.
- *ARM templates*: onboard Sentinel installing OMSGallery solution called SecurityInsights. Sample available here.
- *Terraform*: Using azurerm provider. Sample available here.

Data connectors

There are many (90+) different connectors that you can use to onboard your data sources into Sentinel. We will not cover all of them here, but we provide this summary table for the most typical ones and its coverage across automation methods:

Connector	API	Az Modules	AzSentinel (Community)	ARM
Azure Active Directory	✓	✓	✗	✓
Azure Defender	✓	✓	✓	✓
Azure AD Identity Protection	✓	✓	✗	✓
Microsoft Defender for Identity	✓	✓	✗	✓
Microsoft Cloud App Security	✓	✓	✗	✓
Microsoft Defender for Endpoint	✓	✓	✗	✓
Office 365	✓	✓	✓	✓
Threat Intelligence Platforms	✓	✓	✗	✓
Threat Intelligence TAXII	✓	✗	✓	✗

Although not in the above list, data connectors based on **OMS Solutions** or **DataSources** (Azure Activity, SecurityEvents, Syslog, DNSAnalytics, Windows Firewall, etc.) can be enabled with ARM, PS or API. You can see samples here and here.

Also, data connectors based on Azure **Diagnostics Settings** (Azure Firewall, Azure WAF, Azure DDoS, Azure SQL, etc.) can be enabled with ARM, PS or API. You can see samples here.

Most of the newer connectors being added to Azure Sentinel are based on Azure Functions that use the Log Analytics Data Collector API to send that into Azure Sentinel. This type of connectors is also deployable via ARM templates.

Analytics Rules

Can be deployed using API, PS or ARM templates.

- *API*: all connector types are supported through API. Even if it's not documented in the stable version of the API, you can capture the API request body using developer tools in your browser.
- *PowerShell*: As of today, **AzSentinel** has more feature coverage when it comes to analytics rules. It supports all alert rule types (Scheduled, Microsoft, Fusion, and ML Behavior Analytics) and features like incident grouping. It also has the options to import from file (Import-AzSentinelAlertRule) and attaching a playbook to a rule while creating it (New-AzSentinelAlertRule).

  Az.SecurityInsights doesn't currently support ML Behavior Analytics Rules. Also, being bounded to the stable version API, it has limitations as to what it supports. For example, it doesn't support incident grouping. There are samples scripts that allow for Export/Import of rules (see here).

- *ARM templates:* here you can find sample ARM templates on how to create analytics rules. These samples use the stable version of the SecurityInsights API, so things like incident grouping are not added yet. Changing the *apiVersion* in the template to *2019-01-01-preview* would allow you to add preview features.

Hunting Queries and KQL Functions

Hunting Queries and KQL Functions are just Log Analytics Saved Searches. In the case of Hunting Queries, you just need to specify the category to be "Hunting Queries".

- *PowerShell*: you can create and update Saved Searches via the New-AzOperationalInsightsSavedSearch and Set-AzOperationalInsightsSavedSearch respectively. To update existing saved searches, you will need to provide an *etag* value that should be set to "*" (asterisk).
- *ARM templates*: you can also deploy and update Saved Searches using ARM templates. Like PowerShell, you need to specify an *etag* value when updating. You have samples here and here.

Workbooks and Playbooks

The easiest way to deploy Workbooks and Playbooks is using ARM templates. You can find Workbook ARM template samples here and Playbook samples here. In this other GitHub folder, you will find Workbook samples, but not embedded within an ARM template. These just contain the code that goes inside the *serializedData* property of the workbook (which contains the actual code of the visualizations in the workbook).

For Workbooks, be mindful of two parameters that need to be provided:

- **workbookType**. This value must be set to "**sentinel"** if you want the workbook to show up in *My Workbooks* within the Azure Sentinel interface.
- **workbookSourceId**. This value must be set to the following value, to be associated with the Azure Sentinel workspace:

```
/subscriptions/<subscriptionId>/resourcegroups/<resourceGroup>/providers/microsoft.operationalinsights/workspaces/<workspaceName>
```

Watchlists

As of now, the only method to automate management of watchlists is via the API (in public preview). Here you can see the corresponding documentation. PowerShell support will be added in the future.

Automation Rules

As this feature is still in Preview it lacks API at this stage.

CI/CD pipelines

There are multiple tools that you can use to manage your CI/CD pipelines, but here we will focus mostly on Azure DevOps.

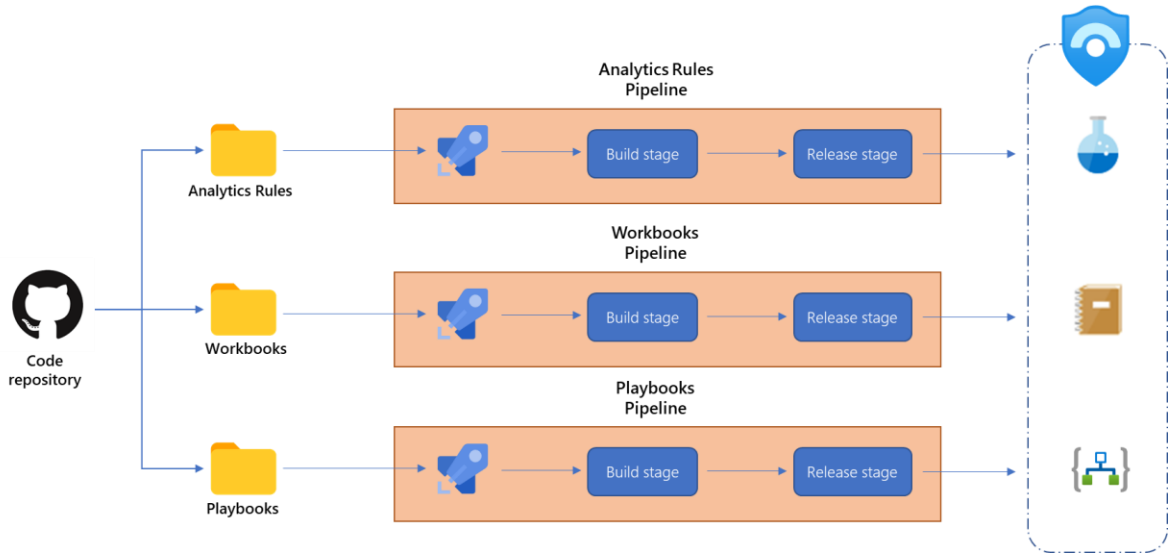
The purpose of CI/CD pipelines is to automate the validation and deployment of infrastructure in an automatic fashion. These pipelines can be triggered based on different events. The most common one is a push event to the code repository where you keep your source code, but there are other options, like triggering when another pipeline has finished.

CI pipelines
Also called **build stage**, the main purpose of CI pipelines is to validate that the code that will be pushed into your environment is valid and won't cause any errors. This validation can be simple, like validating that a given JSON/YAML file has the right format, or it can be more complex, checking if all the relevant properties for a given object are present or if the KQL syntax is correct. You have examples of these two approaches in the two prototypes that were mentioned at the beginning of this chapter: Sentinel as Code and Sentinel DevOps.

CD pipelines
Also called **release stage**, CD pipelines are responsible for deploying the definitions in the source code into the platform, in our case, Azure Sentinel. These pipelines will execute scripts that will take the code in your repository and convert it into objects (Analytics Rules, Connectors, etc.) that can be deployed into your Azure Sentinel workspace. There are also different approaches on how to build these CD pipelines. You can have separate pipelines for each artifact type or one that encompasses all artifacts. Examples can be found on both prototypes mentioned above.

Multi-stage pipelines
There are some CI/CD tools that allow you to create pipelines that include both CI and CD stages. This is the case of Azure DevOps and these are being used in the Azure Sentinel as Code concept.

This picture shows how pipelines could be structured at a high level:



Here we have one multi-stage pipeline for each artifact, that includes both CI (build) and CD (release) stages. These pipelines are triggered whenever a change in the relevant artifact directory is detected.

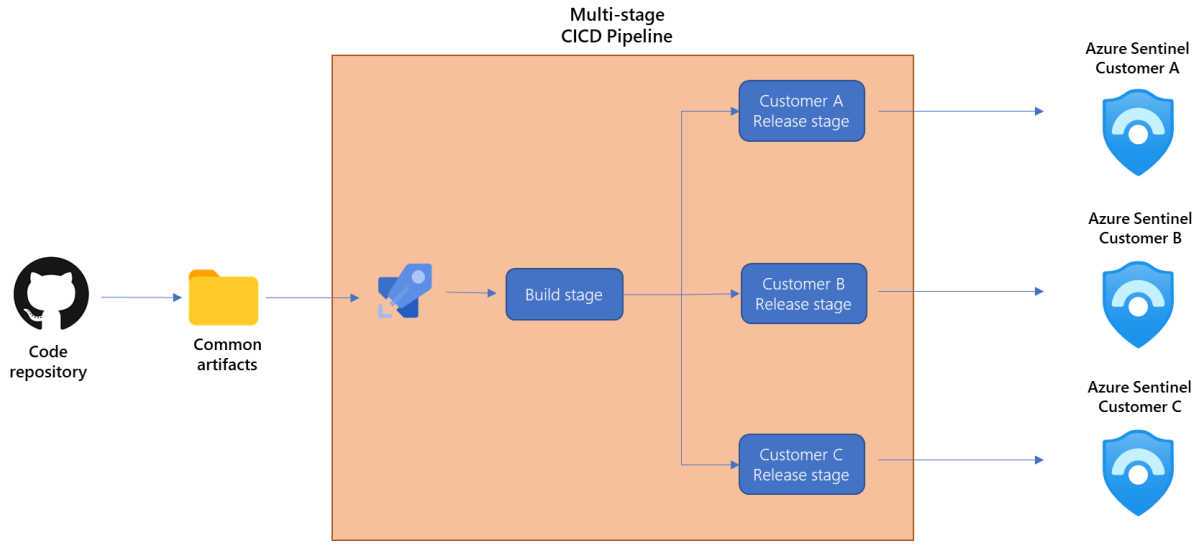
Multi-tenant CI/CD pipelines

An obvious question is how to organize CI/CD pipelines when you're dealing with multiple customers and therefore, multiple workspaces. For an in depth discussion on this topic, please read this article.

The first thing to understand is that using Azure Lighthouse, we can use the **same service principal to access all our customer environments**, we just need to add that service principal as one of the delegated principals in our Lighthouse marketplace offer or ARM template.

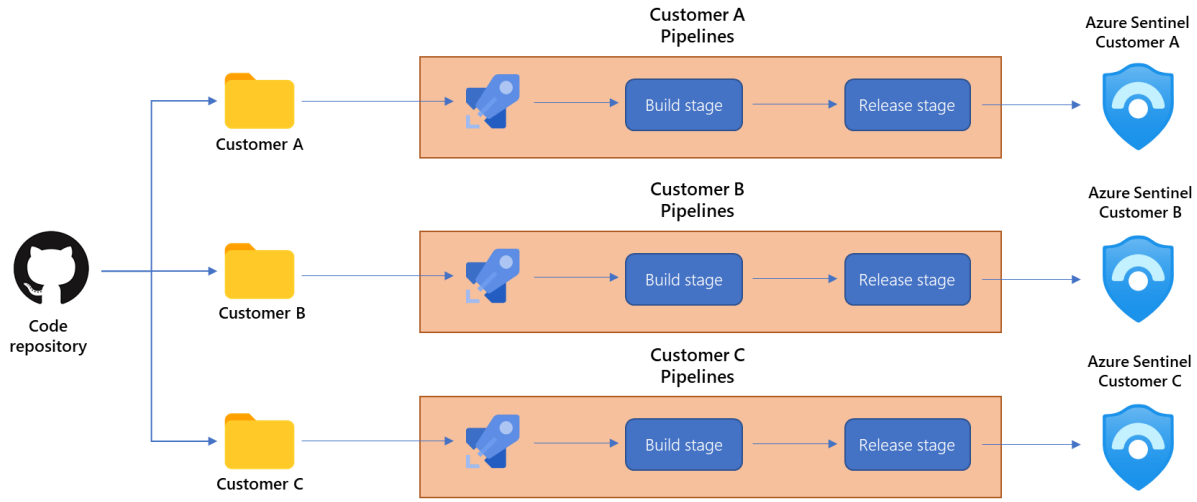
As explained in the article above, the way you organize your pipelines is highly dependent on how your customers are managed. Do they all need the same content (rules, workbooks, etc.) or will each customer have its own

requirements? If all customers will need the same content, you can have a single code repository and a single folder structure where you keep the configurations. It would look like this:



In this case, what we do is create a single pipeline for each artifact type (analytics rules, workbooks, etc.), and that pipeline has multiple release stages, each pointing to a different Azure Sentinel environment.

If your customer will need different content, you could use the following approach:



Here, we have our code repository with separate folders for each customer, and then a set of pipelines (one per artifact) for each.

For more details, refer to this article.

Azure Sentinel All-In-One (MSSP version)

We recently released a new artifact called Azure Sentinel All-In-One which aims to help with fast onboarding of new customers. It is basically a way to have a full Azure Sentinel environment up and running in a matter of seconds, including the enablement of data connectors, rules and even Azure Lighthouse delegation (ARM version only).

There are two versions, PowerShell and ARM template and they automatically perform the following tasks:

- Creates resource group (if given resource group doesn't exist yet)
- Creates the **Azure Lighthouse** registration definition (only MSSP version)
- Creates the **Azure Lighthouse** registration assignments to the resource group that will contain the Azure Sentinel resources (only MSSP version)
- Creates Log Analytics workspace (if given workspace doesn't exist yet)
- Installs Azure Sentinel on top of the workspace (if not installed yet)
- Enables selected Data Connectors from this list:
    - Azure Activity
    - Azure Defender
    - Azure Active Directory Identity Protection
    - Office 365 (SharePoint, Exchange, and Teams)
    - Microsoft Cloud App Security
    - Microsoft Defender for Identity
    - Microsoft Defender for Endpoint
    - Security Events
    - Linux Syslog
    - DNS (Preview)
    - Windows Firewall
- Enables analytics rules for selected Microsoft 1st party products
- Enables Fusion rule and ML Behavior Analytics rules for RDP or SSH (if Security Events or Syslog data sources are selected) – only ARM version
- Enables Scheduled analytics rules that apply to all the enabled connectors. – only ARM version

As mentioned above, there is an **additional version that is targeted for MSSPs**, that, on top of the steps outlined above, also creates the **Azure Lighthouse delegations** need to manage the customer's Sentinel environment.

For more details about this project, visit http://aka.ms/sentinelallinone. Specifically, for the **MSSP version go here**.

Training and community resources

- Ninja Training-The Ninja Training 2021 edition is out! - Microsoft Tech Community
- Azure Sentinel Technical Documentation
- Azure Sentinel Learning Path-https://docs.microsoft.com/en-us/learn/paths/security-ops-sentinel/
- SC-200 – Security Operations Analyst Associate- https://docs.microsoft.com/en-us/learn/certifications/security-operations-analyst/
- Azure Sentinel Tech community – Blogs – https://techcommunity.microsoft.com/t5/azure-sentinel/bg-p/AzureSentinelBlog
- What's new in Azure Sentinel - https://docs.microsoft.com/en-us/azure/sentinel/whats-new
- Top 10 Best Practices for Azure Security (documentation) – https://aka.ms/azuresecuritytop10
- Top 10 Best Practices for Azure Security (video) - https://youtu.be/g0hgtxBDZVE
- Azure Sentinel Documentation site - https://docs.microsoft.com/en-us/azure/sentinel/
- Azure Security Podcast – https://aka.ms/azsecpod
- Azure Security Community Webinars - https://aka.ms/SecurityWebinars
- Azure Sentinel GitHub - https://github.com/Azure/Azure-Sentinel