

- [Azure Logic Apps](#). Create workflows to connect hundreds of services in the cloud and on-premises.
- [Azure Service Bus](#). Connect on-premises and cloud-based applications and services to implement highly secure messaging workflows.
- [Azure Event Grid](#). Connect supported Azure and third-party services while simplifying event-based app development.
- [Azure Functions](#). Simplify complex orchestration problems with an event-driven serverless compute platform.
- [Azure Data Factory](#). Visually integrate data sources to accelerate data transformation and support enterprise workflows.

For information about more Azure integration services, see [Integration Services](#).

Introduction to integration on Azure

If you're new to integration, the best place to start is Microsoft Learn. This free online platform offers videos, tutorials, and hands-on training for various products and services.

The following resources can help you learn the core concepts of integration:

- [Design data integration](#)
- [Integration design for Dynamics 365 solutions](#)
- [Data integrations with Finance and Operations apps](#)
- [Examine business integration for IoT solutions](#)
- [Integrate data with Azure Data Factory or Azure Synapse Pipeline](#)
- [Explore Event Grid integration](#)
- [Architect API integration in Azure](#)

Path to production

After you've covered the fundamentals of integration, the next step is to design your solution.

Design patterns

To explore patterns to incorporate into your design, consult resources in the following areas.

Hybrid systems

- [Cross-cloud scaling—on-premises data](#): See a hybrid app that spans Azure and Azure Stack Hub and uses a single on-premises data source, which is a compliance requirement for some organizations.

Microservice architectures

- [Transactional Outbox pattern with Azure Cosmos DB](#): Implement the Transactional Outbox pattern for reliable messaging between services.
- [Identify microservice boundaries](#): Derive microservices from a domain model when designing your application.
- [Design interservice communication for microservices](#): Use service meshes to make communication between microservices efficient and robust.

Mainframe migration

- [Integrate IBM mainframe and midrange message queues with Azure](#): Use a data-first technique that provides a way for IBM mainframe and midrange message queues to work with Azure services.

Service selectors

The following resources can also help you design your application. Besides providing general information about an integration mechanism or process, each article helps you select an Azure service that best meets your need for that area.

- [Asynchronous messaging options](#): Understand various types of messages and the entities that participate in a messaging infrastructure.
- [Choose between virtual network peering and VPN gateways](#): Explore two ways to connect virtual networks in Azure.
- [Extract, transform, and load \(ETL\)](#): Find out how to gather data that comes from multiple sources in multiple formats, and then transform it and store it.

Specific implementations

To learn about scenario-specific architectures, see the solutions in the following areas.

E-commerce

- [Migrate a web app using Azure APIM](#): Modernize the legacy browser-based software stack of an e-commerce company.

Finance

- [Patterns and implementations for a banking cloud transformation](#): Apply patterns that implement a banking system cloud transformation.

Best practices

These resources can help you spot-check your design against current recommended best practices:

- Azure Event Hubs and Functions can work together in a serverless architecture to process large volumes of data in near real time. For guidance on how to maximize the performance, resiliency, security, observability, and scale of this architecture, see these articles:
 - [Integrate Event Hubs with serverless functions on Azure](#).
 - [Performance and scale for Event Hubs and Azure Functions](#)
 - [Monitor Azure Functions and Event Hubs](#)
- Many integration solutions use Logic Apps to implement business processes. For best practices on building reliable architectures with this service, see [Business continuity and disaster recovery for Azure Logic Apps](#).
- To check whether your Logic Apps implementation aligns with the Azure Security Benchmark version 2.0, see [Azure security baseline for Logic Apps](#).

Suite of baseline implementations

These reference architectures provide baseline implementations for various scenarios:

- [Data analysis workloads for regulated industries](#): Run data analytics workloads that take into account regulatory requirements.
- [Basic enterprise integration on Azure](#): Orchestrate synchronous calls to enterprise back-end systems.
- [Enterprise integration using message broker and events](#): Orchestrate asynchronous calls to enterprise back-end systems by using queues and events.
- [Enterprise business intelligence](#): Move data from an on-premises SQL Server database into Azure Synapse Analytics and transform the data for analysis.
- [Web and mobile front ends](#): Make third-party data available to web users.

Operations guide

Deploying your workload is a significant milestone. After your integration processes are running, your focus can turn to operations. The following materials provide recommendations and reference information to help you continue to meet customer and regulatory demands:

- [About connectors in Azure Logic Apps](#): Learn how to take advantage of the hundreds of connectors that Logic Apps offers.

- [Azure Policy Regulatory Compliance controls for Azure Logic Apps](#): Make Logic Apps compliant with regulatory standards.

Stay current with integration

Azure integration receives improvements on an ongoing basis. To stay on top of recent developments, see [Azure updates](#).

Additional resources

The following resources provide practical recommendations and information for specific scenarios.

Information for Amazon Web Services (AWS)

- [Messaging services on Azure and AWS](#)

Information for Google Cloud professionals

- [Google Cloud to Azure services comparison—messaging and eventing](#)
- [Google Cloud to Azure services comparison—miscellaneous workflow](#)

Integrate Event Hubs with serverless functions on Azure

Azure Event Hubs

Azure Functions

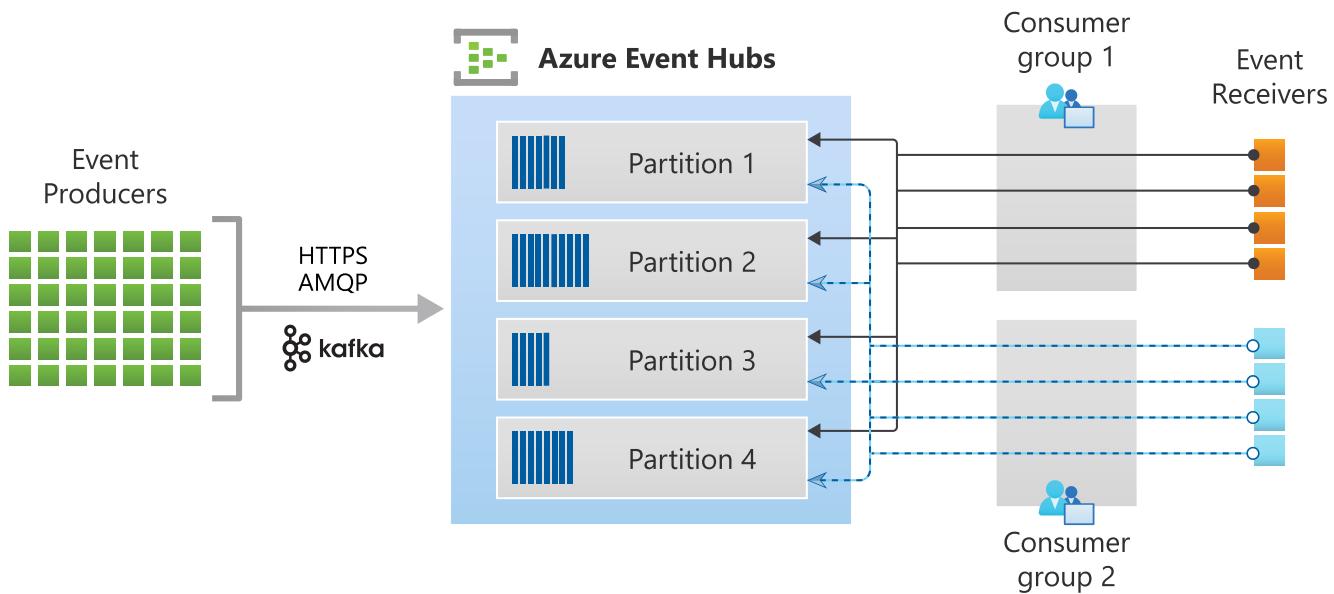
Azure Monitor

Solutions that use Azure Event Hubs together with Azure Functions benefit from a [serverless](#) architecture that is scalable, cost-effective, and capable of processing large volumes of data in near real time. Although these services are commonly used together, there are many features, settings, and intricacies that add complexity to their relationship. This article provides guidance on how to effectively take advantage of this integration by highlighting key considerations and techniques for performance, resiliency, security, observability, and scale.

Event Hubs core concepts

[Azure Event Hubs](#) is a highly scalable event processing service that can receive millions of events per second. Before diving into the patterns and best practices for Azure Functions integration, it's best to understand the fundamental components of Event Hubs.

The following diagram shows the Event Hubs stream processing architecture:

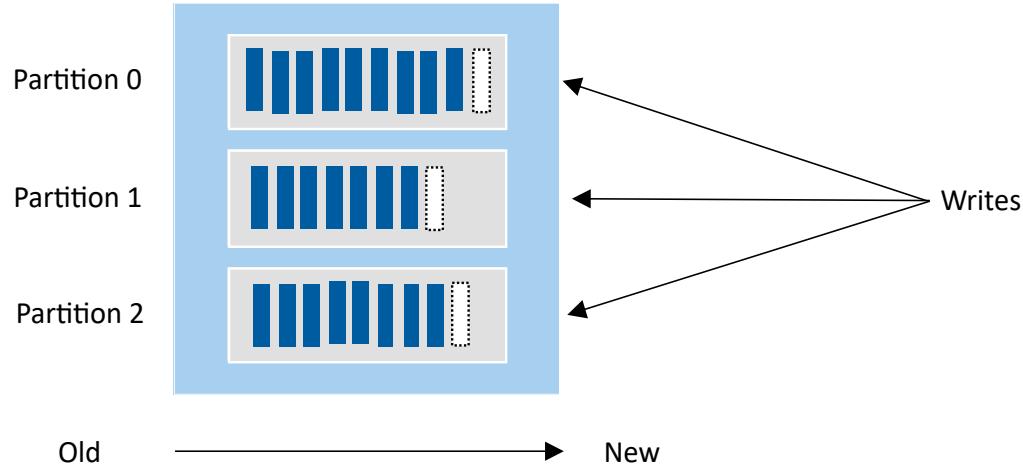


Events

An event is a notification or state change that is represented as a fact that happened in the past. Events are immutable and persisted in an **event hub**, also referred to as a *topic* in [Kafka](#). An event hub is comprised of one or more **partitions**.

Partitions

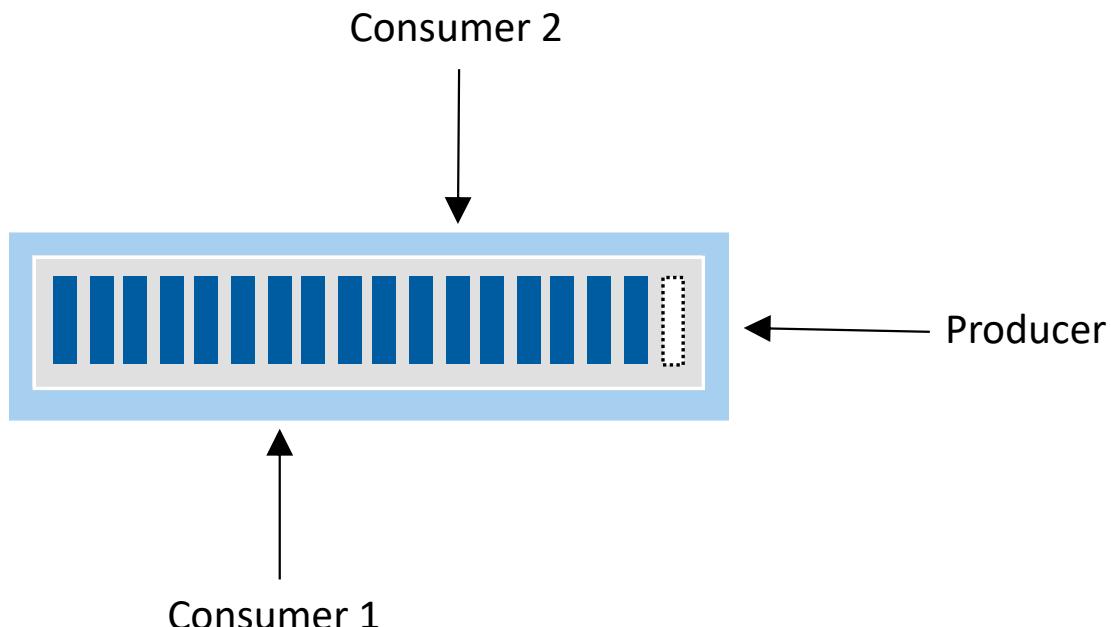
When a partition isn't specified by the sender, received events are distributed across partitions in the event hub. Each event is written in exactly one partition and isn't multi-cast across partitions. Each partition works as a log where records are written in an append-only pattern. The analogy of a *commit log* is frequently used to describe the nature of how events are added to the end of a sequence in a partition.



When more than one partition is used, it allows for parallel logs to be used from within the same event hub. This behavior provides multiple degrees of parallelism and enhances throughput for consumers.

Consumers and consumer groups

A partition can be consumed by more than one consumer, each reading from and managing their own offsets.



Event Hubs has the concept of [consumers groups](#), which enables multiple consuming applications to each have a separate view of the event stream and read the stream independently at their own pace and with their own offsets.

To learn more, see [Deep dive on Event Hubs concepts and features](#).

Consuming events with Azure Functions

Azure Functions supports [trigger](#) and [output](#) bindings for Event Hubs. This section covers how Azure Functions responds to events sent to an event hub event stream using triggers.

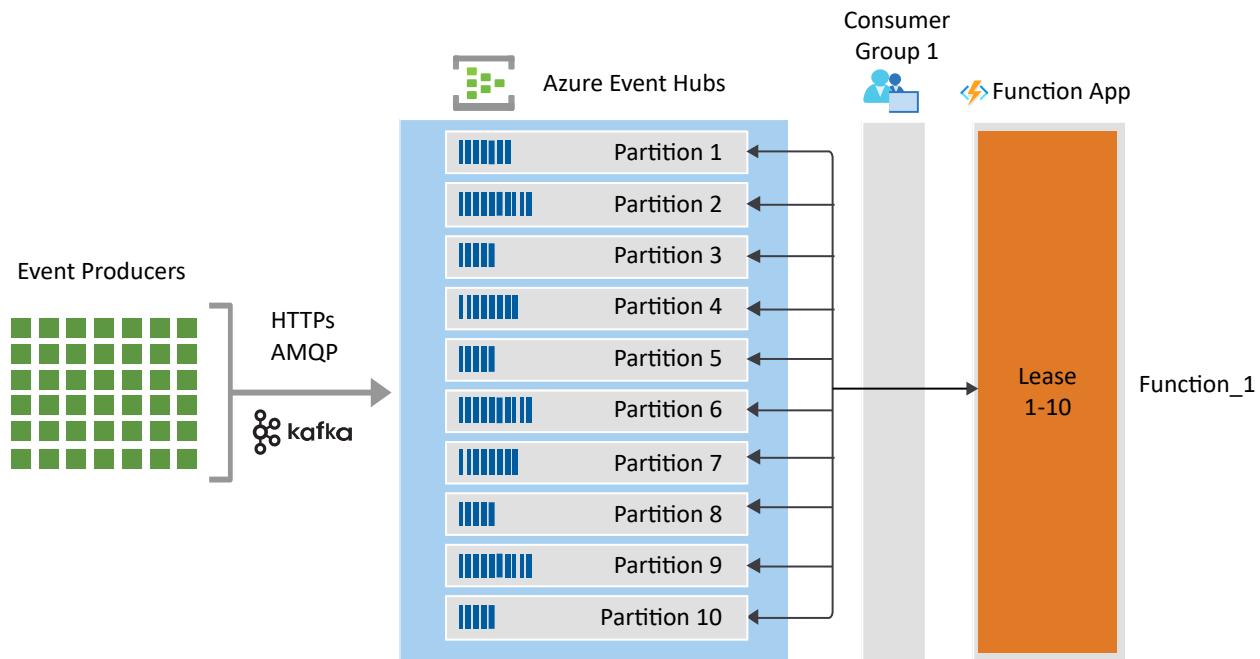
Each instance of an Event Hubs triggered function is backed by a single [EventProcessorHost](#) instance. The trigger (powered by Event Hubs) ensures that only one [EventProcessorHost](#) instance can get a lease on a given partition.

For example, consider an event hub with the following characteristics:

- 10 partitions.
 - 1,000 events distributed across all partitions, with a varying number of messages in each partition.

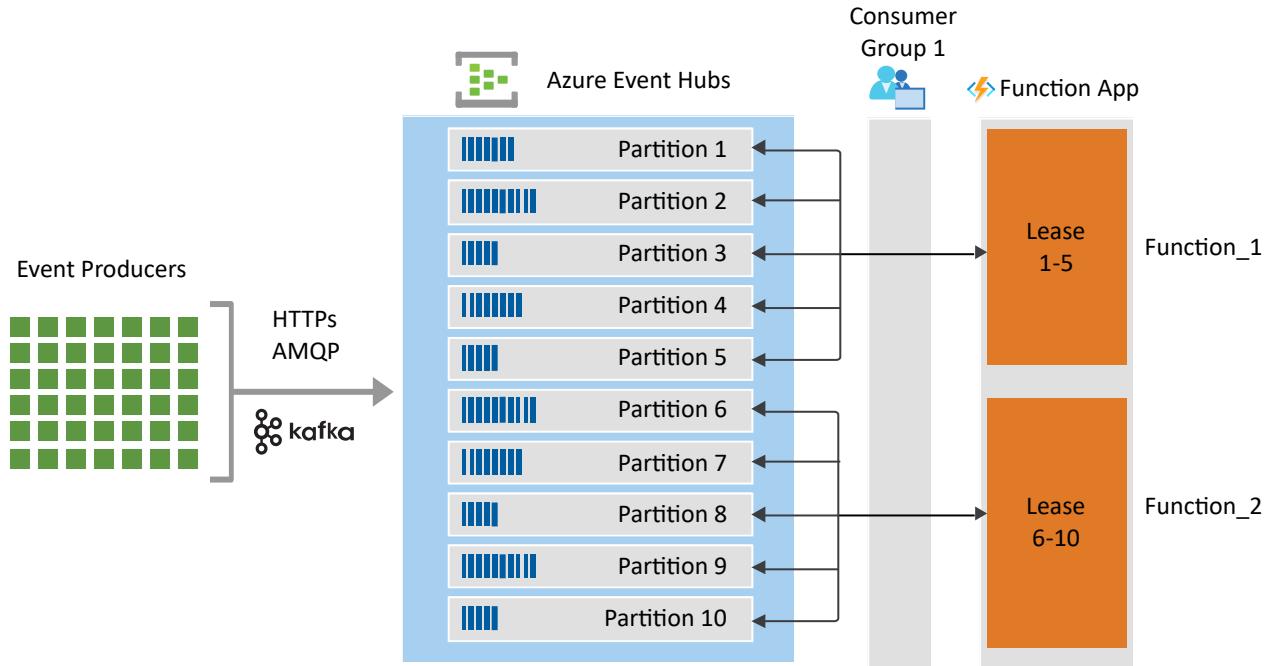
When your function is first enabled, there's only one instance of the function. Let's call the first function instance `Function_1`. `Function_1` has a single instance of `EventProcessorHost` that holds a lease on all 10 partitions. This instance is reading events from partitions 1-10. From this point forward, one of the following happens:

- **New function instances are not needed:** `Function_1` can process all 1,000 events before the Functions scaling logic take effect. In this case, all 1,000 messages are processed by `Function_1`.

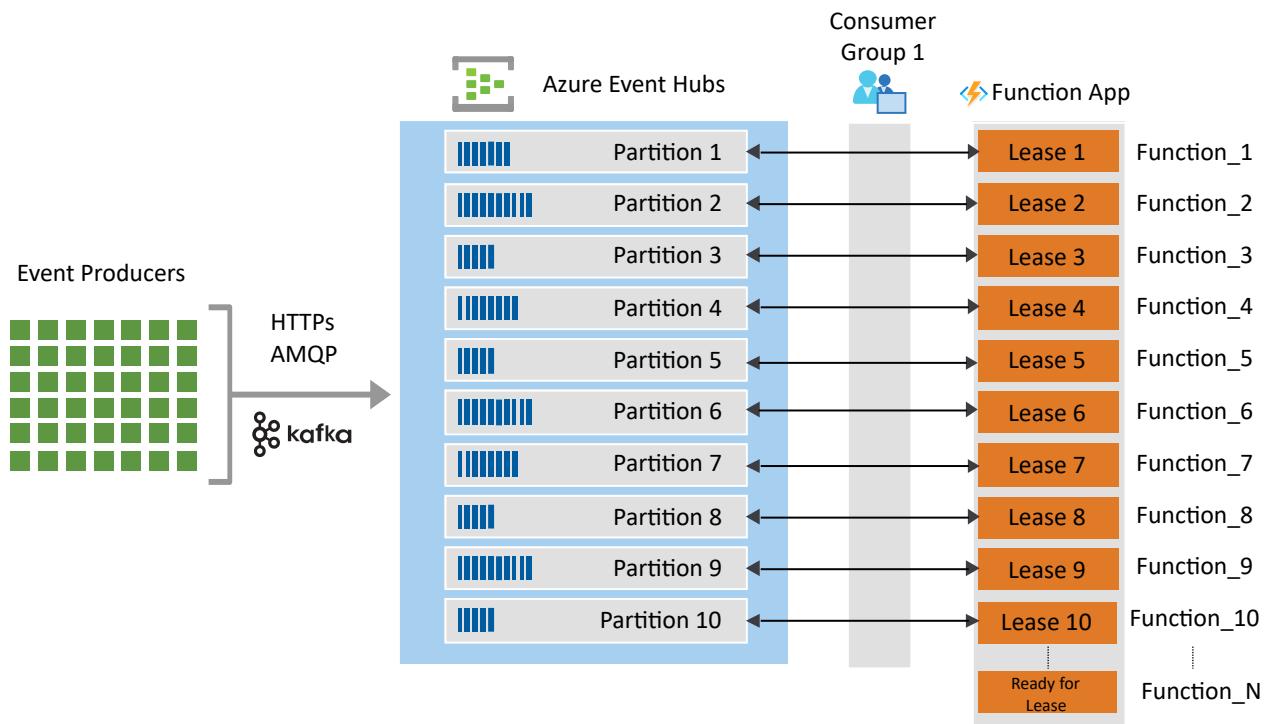


- **An additional function instance is added:** event-based scaling or other automated or manual logic might determine that `Function 1` has more messages than it can process

and then creates a new function app instance (`Function_2`). This new function also has an associated instance of `EventProcessorHost`. As the underlying event hub detects that a new host instance is trying to read messages, it load balances the partitions across the host instances. For example, partitions 1-5 might be assigned to `Function_1` and partitions 6-10 to `Function_2`.



- **N more function instances are added:** event-based scaling or other automated or manual logic determines that both `Function_1` and `Function_2` have more messages than they can process, new `Function_N` function app instances are created. Instances are created to the point where N is equal to or greater than the number of event hub partitions. In our example, Event Hubs again load balances the partitions, in this case across the instances `Function_1`...`Function_10`.



As scaling occurs, N instances can be a number greater than the number of event hub partitions. This situation might occur while event-driven scaling stabilizes instance counts, or because other automated or manual logic created more instances than partitions. In this case, [EventProcessorHost](#) instances will only obtain locks on partitions as they become available from other instances, as at any given time only one function instance from the same consumer group can access/read from the partitions it has locks on.

When all function execution completes (with or without errors), checkpoints are committed to the associated storage account. When checkpointing succeeds, the function will be ready to process a new batch of events.

Dynamic, event-based scaling is possible with Consumption, Flex Consumption, and Premium Azure plans. Kubernetes hosted function apps can also take advantage of the [KEDA scaler for Event Hubs](#). Event-based scaling currently isn't possible when the function app is hosted in a Dedicated (App Service) plan, which requires you to determine the right number of instances based on your workload.

To learn more, see [Azure Event Hubs bindings for Azure Functions](#) and [Azure Event Hubs trigger for Azure Functions](#).

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [David Barkol](#) | Principal Solution Specialist GBB

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

Performance and scale

Performance and scale guidance for Event Hubs and Azure Functions

Azure Event Hubs

Azure Functions

This article provides guidance for optimizing scalability and performance when you use Azure Event Hubs and Azure Functions together in your applications.

Function grouping

Typically, a function encapsulates a unit of work in an event-processing stream. For instance, a function can transform an event into a new data structure or enrich data for downstream applications.

In Functions, a function app provides the execution context for functions. Function app behaviors apply to all functions that the function app hosts. Functions in a function app are deployed together and scaled together. All functions in a function app must be of the same language.

How you group functions into function apps can affect the performance and scaling capabilities of your function apps. You can group according to access rights, deployment, and the usage patterns that invoke your code.

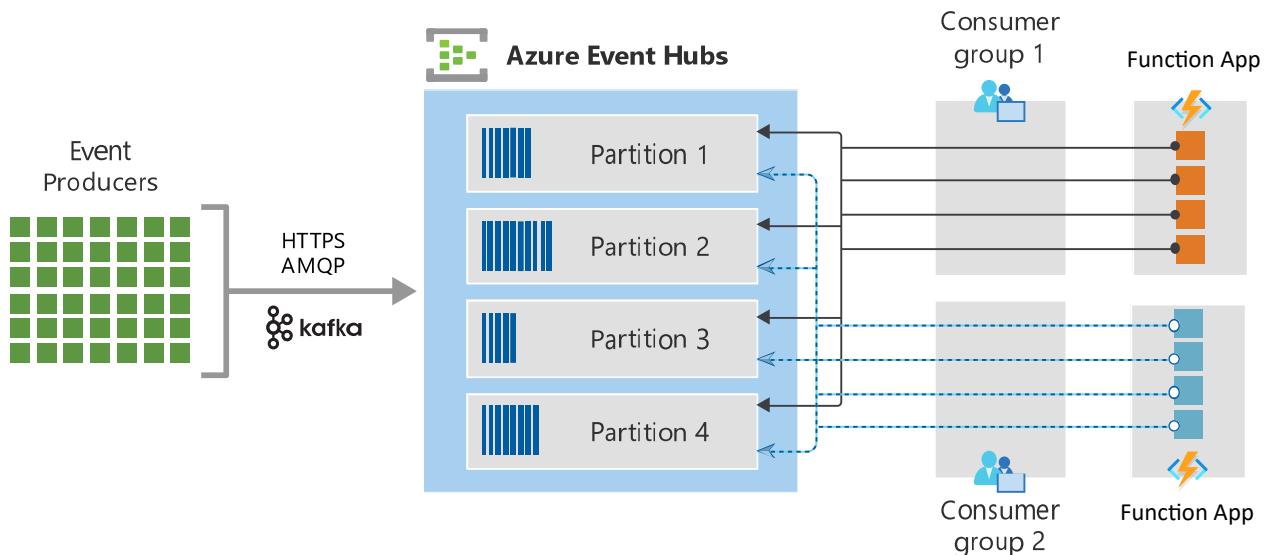
For guidance on Functions best practices for grouping and other aspects, see [Best practices for reliable Azure Functions](#) and [Improve the performance and reliability of Azure Functions](#).

The following list is guidance for grouping functions. The guidance considers storage and consumer group aspects:

- **Host a single function in a function app:** If Event Hubs triggers a function, you can, to reduce contention between that function and other functions, isolate the function in its own function app. Isolation is especially important if the other functions are CPU or memory intensive. This technique helps because each function has its own memory footprint and usage patterns that can directly affect the scaling of the function app that hosts it.
- **Give each function app its own storage account:** Avoid sharing storage accounts between function apps. Also, if a function app uses a storage account, don't use that account for other storage operations or needs. It can be especially important to avoid sharing storage accounts for functions that Event Hubs triggers, because such functions can have a high volume of storage transactions due to checkpointing.

- **Create a dedicated consumer group for each function app:** A consumer group is a view of an event hub. Different consumer groups have different views, which means that the states, positions, and offsets can differ. Consumer groups make it possible for multiple consuming applications to have their own views of the event stream, and to read the stream independently at their own pace and with their own offsets. For more information about consumer groups, see [Features and terminology in Azure Event Hubs](#).

A consumer group has one or more consumer applications associated with it, and a consumer application can use one or more consumer groups. In a stream processing solution, each consumer application equates to a consumer group. A function app is a prime example of a consumer application. The following diagram provides an example of two function apps that read from an event hub, where each app has its own dedicated consumer group:



Don't share consumer groups between function apps and other consumer applications. Each function app should be a distinct application with its own assigned consumer group to ensure offset integrity for each consumer and to simplify dependencies in an event streaming architecture. Such a configuration, along with providing each event hub-triggered function its own function app and storage account, helps set the foundation for optimal performance and scaling.

Function hosting plans

There are several hosting options for function apps and it is important to review their capabilities. For information about these hosting options, see [Azure Functions hosting options](#). Take note of how the options scale.

The Consumption plan is the default. Function apps in the Consumption plan scale independently and are most effective when they avoid long-running tasks.

The Premium and Dedicated plans are often used to host multiple function apps and functions that are more CPU and memory intensive. With the Dedicated plan, you run your functions in an Azure App Service plan at regular App Service plan rates. It's important to note that all the function apps in these plans share the resources that are allocated to the plan. If functions have different load profiles or unique requirements, it's best to host them in different plans, especially in stream processing applications.

Azure Container Apps provides integrated support for developing, deploying, and managing containerized function apps on Azure Functions. This allows you to run your event-driven functions in a fully managed, Kubernetes-based environment with built-in support for open-source monitoring, mTLS, Dapr, and KEDA.

Event Hubs scaling

When you deploy an Event Hubs namespace, there are several important settings that you need to set properly to ensure peak performance and scaling. This section focuses on the Standard tier of Event Hubs and the unique features of that tier that affect scaling when you also use Functions. For more information about Event Hubs tiers, see [Basic vs. Standard vs. Premium vs. Dedicated tiers](#).

An Event Hubs namespace corresponds to a Kafka cluster. For information about how Event Hubs and Kafka relate to one another, see [What is Azure Event Hubs for Apache Kafka](#).

Understanding throughput units (TUs)

In the Event Hubs Standard tier, throughput is classified as the amount of data that enters and is read from the namespace per unit of time. TUs are pre-purchased units of throughput capacity.

TUs are billed on an hourly basis.

All the event hubs in a namespace share the TUs. To properly calculate capacity needs, you must consider all the applications and services, both publishers and consumers. Functions affect the number of bytes and events that are published to and read from an event hub.

The emphasis for determining the number of TUs is on the point of ingress. However, the aggregate for the consumer applications, including the rate at which those events are processed, must also be included in the calculation.

For more information Event Hubs throughput units, see [Throughput units](#).

Scale up with Auto-inflate

Auto-inflate can be enabled on an Event Hubs namespace to accommodate situations in which the load exceeds the configured number of TUs. Using Auto-inflate prevents throttling of your application and helps ensure that processing, including the ingesting of events, continues without disruption. Because the TU setting affects costs, using Auto-inflate helps address concerns about overprovisioning.

Auto-inflate is a feature of Event Hubs that's often confused with autoscale, especially in the context of serverless solutions. However, Auto-inflate, unlike autoscale, doesn't scale-down when added capacity is no longer needed.

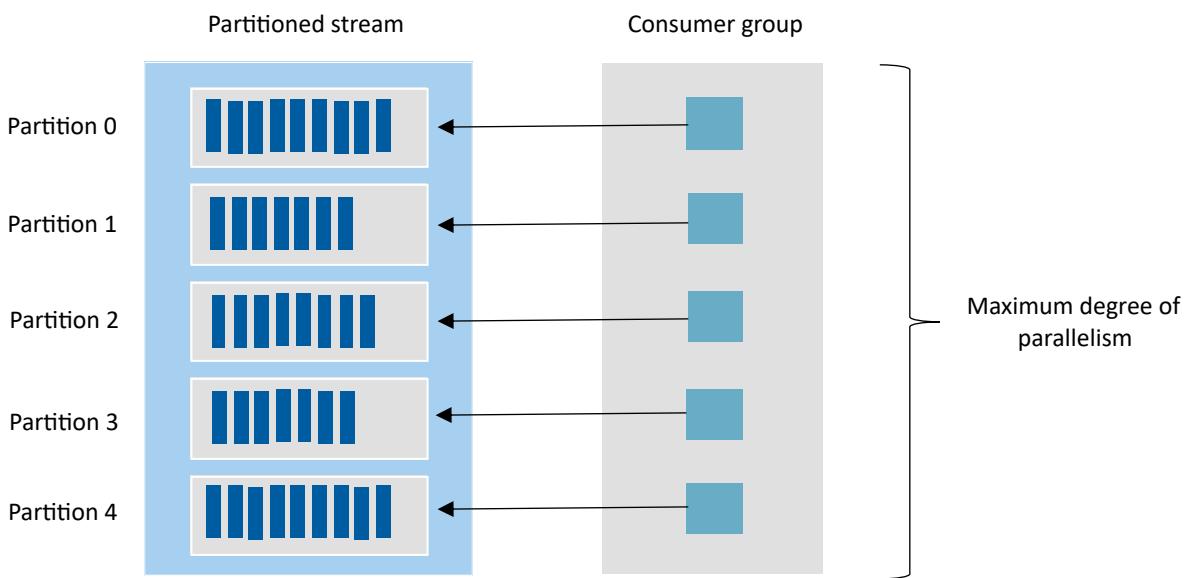
If the application needs capacity that exceeds the maximum allowed number of TUs, consider using Event Hubs [Premium tier](#) or [Dedicated tier](#).

Partitions and concurrent functions

When an event hub is created, the number of [partitions](#) must be specified. The partition count remains fixed and can't be changed except from the Premium and Dedicated tiers. When Event Hubs triggers functions apps, it's possible that the number of concurrent instances can equal the number of partitions.

In Consumption and Premium hosting plans, the function app instances scale out dynamically to meet the number of partitions, if needed. The Dedicated hosting plan runs functions in an App Service plan and requires that you manually configure your instances or set up an autoscale scheme. For more information, see [Dedicated hosting plans for Azure Functions](#).

Ultimately, a one-to-one relationship between the number of partitions and function instances, or consumers, is the ideal target for maximum throughput in a stream processing solution. To achieve optimal parallelism, have multiple consumers in a consumer group. For Functions, this objective translates to many instances of a function in the plan. The result is referred to as *partition-level parallelism* or the *maximum degree of parallelism*, as shown in the following diagram:



It might seem to make sense to configure as many partitions as possible to achieve maximum throughput and to account for the possibility of a higher volume of events. However, there are several important factors to consider when you configure many partitions:

- **More partitions can lead to more throughput:** Because the degree of parallelism is the number of consumers (function instances), the more partitions there are, the higher the concurrent throughput can be. This fact is important when you share a designated number of TUs for an event hub with other consumer applications.
- **More functions can require more memory:** As the number of function instances increases, so does the memory footprint of resources in the plan. At some point, too many partitions can deteriorate performance for consumers.
- **There's a risk of back pressure from downstream services:** As more throughput is generated, you run the risk of overwhelming downstream services or receiving back pressure from them. Consumer fan-out must be accounted for when considering the consequences to surrounding resources. Possible consequences included throttling from other services, network saturation, and other forms of resource contention.
- **Partitions can be sparsely populated:** The combination of many partitions and a low volume of events can lead to data that's sparsely distributed across partitions. Instead, a smaller number of partitions can provide better performance and resource usage.

Availability and consistency

When a partition key or ID isn't specified, Event Hubs routes an incoming event to the next available partition. This practice provides high availability and helps increase throughput for consumers.

When ordering of a set of events is required, the event producer can specify that a particular partition is to be used for all the events of the set. The consumer application that reads from the partition receives the events in proper order. This tradeoff provides consistency but

compromises availability. Don't use this approach unless the order of events must be preserved.

For Functions, ordering is achieved when events are published to a particular partition and an Event Hubs triggered function obtains a lease to the same partition. Currently, the ability to configure a partition with the Event Hubs output binding isn't supported. Instead, the best approach is to use one of the Event Hubs SDKs to publish to a specific partition.

For more information about how Event Hubs supports availability and consistency, see [Availability and consistency in Event Hubs](#).

Event Hubs trigger

This section focuses on the settings and considerations for optimizing functions that Event Hubs triggers. Factors include batch processing, sampling, and related features that influence the behavior of an event hub trigger binding.

Batching for triggered functions

You can configure functions that an event hub triggers to process a batch of events or one event at a time. Processing a batch of events can be more efficient when it reduces some of the overhead of function invocations. Unless you need to process only a single event, your function should be configured to process multiple events when invoked.

Enabling batching for the Event Hubs trigger binding varies between languages:

- JavaScript, Python, and other languages enable batching when the **cardinality** property is set to **many** in the `function.json` file for the function.
- In C#, **cardinality** is automatically configured when an array is designated for the type in the **EventHubTrigger** attribute.

For more information about how batching is enabled, see [Azure Event Hubs trigger for Azure Functions](#).

Trigger settings

Several configuration settings in the `host.json` file play a key role in the performance characteristics of the Event Hubs trigger binding for Functions:

- **maxEventBatchSize**: This setting represents the maximum number of events that the function can receive when it's invoked. If the number of events received is less than this

amount, the function is still invoked with as many events as are available. You can't set a minimum batch size.

- **prefetchCount:** The prefetch count is one of the most important settings when you optimize for performance. The underlying AMQP channel references this value to determine how many messages to fetch and cache for the client. The prefetch count should be greater than or equal to the **maxEventBatchSize** value and is commonly set to a multiple of that amount. Setting this value to a number less than the **maxEventBatchSize** setting can hurt performance.
- **batchCheckpointFrequency:** As your function processes batches, this value determines the rate at which checkpoints are created. The default value is 1, which means that there's a checkpoint whenever a function successfully processes a single batch. A checkpoint is created at the partition level for each reader in the consumer group. For information about how this setting influences replays and retries of events, see [Event hub triggered Azure function: Replays and Retries \(blog post\)](#).

Do several performance tests to determine the values to set for the trigger binding. We recommend that you change settings incrementally and measure consistently to fine-tune these options. The default values are a reasonable starting point for most event processing solutions.

Checkpointing

Checkpoints mark or commit reader positions in a partition event sequence. It's the responsibility of the Functions host to checkpoint as events are processed and the setting for the batch checkpoint frequency is met. For more information about checkpointing, see [Features and terminology in Azure Event Hubs](#).

The following concepts can help you understand the relationship between checkpointing and the way that your function processes events:

- **Exceptions still count towards success:** If the function process doesn't crash while processing events, the completion of the function is considered successful, even if exceptions occurred. When the function completes, the Functions host evaluates **batchCheckpointFrequency**. If it's time for a checkpoint, it creates one, regardless of whether there were exceptions. The fact that exceptions don't affect checkpointing shouldn't affect your proper use of exception checking and handling.
- **Batch frequency matters:** In high-volume event streaming solutions, it can be beneficial to change the **batchCheckpointFrequency** setting to a value greater than 1. Increasing this value can reduce the rate of checkpoint creation and, as a consequence, the number of storage I/O operations.
- **Replays can happen:** Each time a function is invoked with the Event Hubs trigger binding, it uses the most recent checkpoint to determine where to resume processing. The offset

for every consumer is saved at the partition level for each consumer group. Replays happen when a checkpoint doesn't occur during the last invocation of the function, and the function is invoked again. For more information about duplicates and deduplication techniques, see [Idempotency](#).

Understanding checkpointing becomes critical when you consider best practices for error handling and retries, a topic that's discussed later in this article.

Telemetry sampling

Functions provides built-in support for Application Insights, an extension of Azure Monitor that provides application performance monitoring capabilities. With this feature, you can log information about function activities, performance, runtime exceptions, and more. For more information, see [Application Insights overview](#).

This capability offers key configuration choices that affect performance. Some of the notable settings and considerations for monitoring and performance are:

- **Enable telemetry sampling:** For high-throughput scenarios, you should evaluate the amount of telemetry and information that you need. Consider using the [telemetry sampling](#) feature in Application Insights to avoid degrading the performance of your function with unnecessary telemetry and metrics.
- **Configure aggregation settings:** Examine and configure the frequency of aggregating and sending data to Application Insights. This configuration setting is in the [host.json](#) file along with many other sampling and logging related options. For more information, see [Configure the aggregator](#).
- **Disable AzureWebJobDashboard:** For apps that target version 1.x of the Functions runtime, this setting stores the connection string to a storage account that the Azure SDK uses to retain logs for the WebJobs dashboard. If Application Insights is used instead of the WebJobs dashboard, then this setting should be removed. For more information, see [AzureWebJobsDashboard](#).

When Application Insights is enabled without sampling, all telemetry is sent. Sending data about all events can have a detrimental effect on the performance of the function, especially under high-throughput event streaming scenarios.

Taking advantage of sampling and continually assessing the appropriate amount of telemetry needed for monitoring is crucial for optimum performance. Telemetry should be used for general platform health evaluation and for occasional troubleshooting, not to capture core business metrics. For more information, see [Configure sampling](#).

Output binding

Use the [Event Hubs output binding for Azure Functions](#) to simplify publishing to an event stream from a function. The benefits of using this binding include:

- **Resource management:** The binding handles both the client and connection lifecycles for you, and reduces the potential for issues that can arise with port exhaustion and connection pool management.
- **Less code:** The binding abstracts the underlying SDK and reduces the amount of code that you need to publish events. It helps you write and maintain code with fewer boilerplate lines.
- **Batching:** For several languages, batching is supported to efficiently publish to an event stream. Batching can improve performance and help streamline the code that sends the events.

We strongly recommend that you review the list of [Languages that Functions supports](#) and the developer guides for those languages. The **Bindings** section for each language provides detailed examples and documentation.

Batching when publishing events

If your function only publishes a single event, configuring the binding to return a value is a common approach that's helpful if the function execution always ends with a statement that sends the event. This technique should only be used for synchronous functions that return only one event.

Batching is encouraged to improve performance when sending multiple events to a stream. Batching allows the binding to publish events in the most efficient possible way.

Support for using the output binding to send multiple events to Event Hubs is available in C#, Java, Python, and JavaScript.

Output multiple events with the In-process model (C#)

Use the **ICollector** and **IAsyncCollector** types when you send multiple events from a function in C#.

- The **ICollector<T>.Add()** method can be used in both synchronous and asynchronous functions. It executes the add operation as soon as it's called.
- The **IAsyncCollector<T>.AddAsync()** method prepares the events to be published to the event stream. If you write an asynchronous function, you should use **IAsyncCollector** to better manage the published events.

For examples of using C# to publish single and multiple events, see [Azure Event Hubs output binding for Azure Functions](#).

Output multiple events with the Isolated worker model (C#)

Depending on the Functions runtime version, the Isolated worker model will support different types for the parameters that are passed to the output binding. For multiple events, an array is used to encapsulate the set. It is recommended to review the output binding attributes and usage details for the Isolated model and to make note of the differences between the extension versions.

Throttling and back pressure

Throttling considerations apply to output bindings, not only for Event Hubs but also for Azure services such as [Azure Cosmos DB](#). It's important to become familiar with the limits and quotas that apply to those services and to plan accordingly.

To handle downstream errors with the In-process model, you can wrap `AddAsync` and `FlushAsync` in an exception handler for .NET Functions in order to catch exceptions from `IAsyncCollector`. Another option is to use the Event Hubs SDKs directly instead of using output bindings.

If you are leveraging the Isolated model for functions, then structured exception handling should be used to responsibly catch exceptions when returning the output values.

Function code

This section covers the key areas that must be considered when writing code to process events in a function that Event Hubs triggers.

Asynchronous programming

We recommend that you write your function to [use async code and avoid blocking calls](#), especially when I/O calls are involved.

Here are guidelines that you should follow when you write a function to process asynchronously:

- **All asynchronous or all synchronous:** If a function is configured to run asynchronously, all the I/O calls should be asynchronous. In most cases, partially asynchronous code is worse than code that's entirely synchronous. Choose either asynchronous or synchronous, and stick with the choice all the way through.
- **Avoid blocking calls:** Blocking calls return to the caller only after the call completes, in contrast to asynchronous calls that return immediately. An example in C# would be calling `Task.Result` or `Task.Wait` on an asynchronous operation.

More about blocking calls

Using blocking calls for asynchronous operations can lead to thread-pool starvation and cause the function process to crash. The crash happens because a blocking call requires another thread to be created to compensate for the original call that's now waiting. As a result, it requires twice as many threads to complete the operation.

Avoiding this *sync over async* approach is especially important when Event Hubs is involved, because a function crash doesn't update the checkpoint. The next time the function is invoked it could end up in this cycle and appear to be stuck or to move along slowly as function executions eventually time out.

Troubleshooting this phenomenon usually starts with reviewing the trigger settings and running experiments that can involve increasing the partition count. Investigations can also lead to changing several of the batching options such as the max batch size or prefetch count. The impression is that it's a throughput problem or configuration setting that just needs to be tuned accordingly. However, the core problem is in the code itself and must be addressed there.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributor.

Principal author:

- [David Barkol](#) | Principal Solution Specialist GBB

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

Before continuing, consider reviewing these related articles:

- [Monitor executions in Azure Functions](#)
- [Azure Functions reliable event processing](#)
- [Designing Azure Functions for identical input](#)
- [ASP.NET Core async guidance](#).
- [Azure Event Hubs trigger for Azure Functions](#)

[Resilient Event Hubs and Functions design](#)

Resilient Event Hubs and Functions design

08/21/2025

Error handling, designing for idempotency and managing retry behavior are a few of the critical measures you can take to ensure Event Hubs triggered functions are resilient and capable of handling large volumes of data. This article covers these crucial concepts and makes recommendations for serverless event-streaming solutions.

Azure provides three main messaging services that can be used with Azure Functions to support a wide range of unique, event-driven scenarios. Because of its partitioned consumer model and ability to ingest data at a high rate, Azure Event Hubs is commonly used for event streaming and big data scenarios. For a detailed comparison of Azure messaging services, see [Choose between Azure messaging services - Event Grid, Event Hubs, and Service Bus](#).

Streaming benefits and challenges

Understanding the benefits and drawbacks of streams helps you appreciate how a service like [Event Hubs](#) operates. You also need this context when making impactful architectural decisions, troubleshooting issues, and optimizing for performance. Consider the following key concepts about solutions featuring both Event Hubs and Functions:

- **Streams are not queues:** Event Hubs, Kafka, and other similar offerings that are built on the partitioned consumer model don't intrinsically support some of the principal features in a message broker like [Service Bus](#). Perhaps the biggest indicator of these differences is the fact that reads are **non-destructive**. This ensures that the data read by the Functions host remains available afterwards. Instead, messages are immutable and remain for other consumers to read, including potentially the same consumer reading it again. For this reason, solutions that implement patterns such as [competing consumers](#) might be better served with a message broker such as Service Bus.
- **Missing inherent dead-letter support:** A dead-letter channel is not a native feature in Event Hubs or Kafka. Often, the *concept* of dead-lettering is integrated into a streaming solution to account for data that cannot be processed. This functionality is intentionally not an innate element in Event Hubs and is only added on the consumer side to manufacture a similar behavior or effect. If you need dead-letter support, you should potentially review your choice of a streaming message service.
- **A unit of work is a partition:** In a traditional message broker, a unit of work is a single message. In a streaming solution, a partition is often considered the unit of work. If each event in an event hub is treated as a distinct message requiring order processing or

financial transaction handling, it suggests an opportunity to explore a more suitable messaging service for optimal performance or processing.

- **No server-side filtering:** One of the reasons Event Hubs is capable of tremendous scale and throughput is due to the low overhead on the service itself. Features like server-side filtering, indexes, and cross-broker coordination aren't part of the architecture of Event Hubs. Functions are occasionally used to filter events by routing them to other event hubs based on the contents in the body or header. This approach is common in event streaming but comes with the caveat that the initial function reads and evaluates each event.
- **Every reader must read all data:** Since server-side filtering is unavailable, a consumer sequentially reads all the data in a partition. This includes data that may not be relevant or could even be malformed. Several options and strategies can be used to compensate for these challenges, which are covered later in this section.

These design decisions allow Event Hubs to support a significant influx of events and provide a resilient service for consumers to read from. Each consumer application is tasked with the responsibility of maintaining their own, client-side offsets or cursor to those events. The low overhead makes Event Hubs a cost-effective option for event streaming.

Idempotency

One of the core tenets of Azure Event Hubs is the concept of at-least once delivery. This approach ensures that events are always delivered. It also means that events can be received more than once, even repeatedly, by consumers such as a function. For this reason, it's important that an event hub triggered function supports the [idempotent consumer](#) pattern.

Working under the assumption of at-least once delivery, especially within the context of an event-driven architecture, is a responsible approach for reliably processing events. Your function must be idempotent so that the outcome of processing the same event multiple times is the same as processing it once.

Duplicate events

There are several different scenarios that could result in duplicate events being delivered to a function:

- **Checkpointing:** If the Azure Functions host crashes or the threshold set for the [batch checkpoint frequency](#) is not met, a checkpoint is not created. As a result, the offset for the consumer is not advanced and the next time the function is invoked, it will resume from

the last checkpoint. It is important to note that checkpointing occurs at the partition level for each consumer.

- **Duplicate events published:** Many techniques can reduce the chances of the same event being published to a stream, but the consumer is still responsible for handling duplicates idempotently.
- **Missing acknowledgments:** In some situations, an outgoing request to a service may be successful, however, an acknowledgment (ACK) from the service is never received. This perception might result in the belief that the outgoing call failed and initiate a series of retries or other outcomes from the function. In the end, duplicate events could be published, or a checkpoint is not created.

Deduplication techniques

Designing your functions for [identical input](#) should be the default approach taken when paired with the Event Hub trigger binding. You should consider the following techniques:

- **Looking for duplicates:** Before processing, take the necessary steps to validate that the event should be processed. In some cases, this requires an investigation to confirm that it is still valid. It could also be possible that handling the event is no longer necessary due to data freshness or logic that invalidates the event.
- **Design events for idempotency:** By providing additional information within the payload of the event, it is possible to ensure that processing it multiple times does not have any detrimental effects. Take the example of an event that includes an amount to withdrawal from a bank account. If not handled responsibly, it is possible that it could decrement the balance of an account multiple times. However, if the same event includes the updated balance to the account, it could be used to perform an upsert operation to the bank account balance. This event-carried state transfer approach occasionally requires coordination between producers and consumers and should be used when it makes sense to participating services.

Error handling and retries

Error handling and retries are a few of the most important qualities of distributed, event-driven applications, and Functions are no exception. For event streaming solutions, the need for proper error handling support is crucial, as thousands of events can quickly turn into an equal number of errors if they are not handled correctly.

Error handling guidance

Without error handling, it can be tricky to implement retries, detect runtime exceptions, and investigate issues. Every function should have at least some level of error handling. A few recommended guidelines are:

- **Use Application Insights:** Enable and use Application Insights to log errors and monitor the health of your functions. Be mindful of the configurable sampling options for scenarios that process a high volume of events.
- **Add structured error handling:** Apply the appropriate error handling constructs for each programming language to catch, log, and detect anticipated and unhandled exceptions in your function code. For instance, use a try/catch block in C#, Java and JavaScript and take advantage of the [try and except ↗](#) blocks in Python to handle exceptions.
- **Logging:** Catching an exception during execution provides an opportunity to log critical information that could be used to detect, reproduce, and fix issues reliably. Log the exception, not just the message, but the body, inner exception and other useful artifacts that are helpful later.
- **Do not catch and ignore exceptions:** One of the worst things you can do is catch an exception and do nothing with it. If you catch a generic exception, log it somewhere. If you don't log errors, it's difficult to investigate bugs and reported issues.

Retries

Implementing retry logic in an event streaming architecture can be complex. Supporting cancellation tokens, retry counts and exponential back off strategies are just a few of the considerations that make it challenging. Fortunately, Functions provides [retry policies](#) that can make up for many of these tasks that you would typically code yourself.

Several important factors that must be considered when using the retry policies with the Event Hub binding, include:

- **Avoid indefinite retries:** When the [max retry count](#) setting is set to a value of -1, the function retries indefinitely. In general, indefinite retries should be used sparingly with Functions and almost never with the Event Hub trigger binding.
- **Choose the appropriate retry strategy:** A [fixed delay](#) strategy may be optimal for scenarios that receive back pressure from other Azure services. In these cases, the delay can help avoid throttling and other limitations encountered from those services. The [exponential back off](#) strategy offers more flexibility for retry delay intervals and is commonly used when integrating with third-party services, REST endpoints, and other Azure services.

- **Keep intervals and retry counts low:** When possible, try to maintain a retry interval shorter than one minute. Also, keep the maximum number of retry attempts to a reasonably low number. These settings are especially pertinent when running in the Functions Consumption plan.
- **Circuit breaker pattern:** A transient fault error from time to time is expected and a natural use case for retries. However, if a significant number of failures or issues are occurring during the processing of the function, it may make sense to stop the function, address the issues and restart later.

An important takeaway for the retry policies in Functions is that it is a best effort feature for reprocessing events. It does not substitute the need for error handling, logging, and other important patterns that provide resiliency to your code.

Strategies for failures and corrupt data

There are several noteworthy approaches that you can use to compensate for issues that arise due to failures or bad data in an event stream. Some fundamental strategies are:

- **Stop sending and reading:** To fix the underlying issue, pause the reading and writing of events. The benefit of this approach is that data won't be lost, and operations can resume after a fix is rolled out. This approach may require a circuit-breaker component in the architecture and possibly a notification to the affected services to achieve a pause. In some cases, stopping a function may be necessary until the issues are resolved.
- **Drop messages:** If messages aren't important or are considered non-mission critical, consider moving on and not processing them. This approach doesn't work for scenarios that require strong consistency such as recording moves in a chess match or finance-based transactions. Error handling inside of a function is recommended for catching and dropping messages that can't be processed.
- **Retry:** There are many situations that may warrant the reprocessing of an event. The most common scenario would be a transient error encountered when calling another service or dependency. Network errors, service limits and availability, and strong consistency are perhaps the most frequent use cases that justify reprocessing attempts.
- **Dead letter:** The idea here is to publish the event to a different event hub so that the existing flow is not interrupted. The perception is that it is moved off the hot path and can be dealt with later or by a different process. This solution is used frequently for handling poisoned messages or events. Each function configured with a different consumer group will still encounter bad or corrupt data in their stream and must handle it responsibly.

- **Retry and dead letter:** The combination of numerous retry attempts before ultimately publishing to a dead letter stream once a threshold is met, is another familiar method.
- **Use a schema registry:** A schema registry can be used as a proactive tool to help improve consistency and data quality. The [Azure Schema Registry](#) can support the transition of schemas along with versioning and different compatibility modes as schemas evolve. At its core, the schema serves as a contract between producers and consumers, which could reduce the possibility of invalid or corrupt data being published to the stream.

In the end, there isn't a perfect solution and the consequences and tradeoffs of each of the strategies needs to be thoroughly examined. Based on the requirements, using several of these techniques together may be the best approach.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [David Barkol](#) | Principal Solution Specialist GBB

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

Before continuing, consider reviewing these related articles:

- [Azure Functions reliable event processing](#)
- [Designing Azure Functions for identical input](#)
- [Azure Functions error handling and retry guidance](#)

Security

Secure Azure Functions with Event Hubs

Article • 12/17/2024

When configuring access to resources in Azure, you should apply fine-grained control over permissions to resources. Access to these resources should be based on *need to know* and *least privilege* security principles to make sure that clients can only perform the limited set of actions granted to them.

Authorizing Access to Event Hubs

Authorizing access to Azure Event Hubs resources can be done using the following security constructs:

- **Microsoft Entra ID:** Microsoft Entra ID provides role-based access control (RBAC) for granular control over a client's access to Event Hubs resources. Based on roles and permissions granted, Microsoft Entra ID will authorize requests using an OAuth 2.0 access token.
- **Shared access signature:** A shared access signature (SAS) offers the ability to protect Event Hubs resources based on authorization rules. You define authorization policies by selecting one or more [policy rules](#), such as the ability to send messages, listen to messages, and manage the entities in the namespace.

Shared access signature considerations

When using a shared access signature with Azure Functions and Event Hubs, the following considerations should be reviewed:

- **Avoid the Manage right:** In addition to being able to manage the entities in an Event Hubs namespace, the Manage right includes both Send and Listen rights. Ideally, a function app should only be granted a combination of the Send and Listen rights, based on the actions they perform.
- **Don't use the default Manage rule:** Avoid using the default policy rule named `RootManageSharedAccessKey` unless it's needed by your function app, which should be an uncommon scenario. Another caveat to this default rule is that it's created at the namespace level and grants permissions to all underlying event hubs.
- **Review shared access policy scopes:** Shared access policies can be created at the namespace level and per event hub. Consider creating granular access policies that

are tailored for each client to limit their range and permissions.

Managed identity

An Active Directory identity can be assigned to a managed resource in Azure such as a function app or web app. Once an identity is assigned, it has the capabilities to work with other resources that use Microsoft Entra ID for authorization, much like a [service principal](#).

Function apps can be assigned a [managed identity](#) and take advantage of identity-based connections for a subset of services, including Event Hubs. Identity-based connections provide support for both the trigger and output binding extensions and must use the [Event Hubs extension 5.x and higher](#) for support.

Network

By default, Event Hubs namespaces are accessible from the internet, so long as the request comes with valid authentication and authorization. There are three options for limiting network access to Event Hubs namespaces:

- [Allow access from specific IP addresses](#)
- [Allow access from specific virtual networks \(service endpoints\)](#)
- [Allow access via private endpoints](#)

In all cases, it's important to note that at least one IP firewall rule or virtual network rule for the namespace is specified. Otherwise, if no IP address or virtual network rule is specified, the namespace is accessible over the public internet (using the access key).

Azure Functions can be configured to consume events from or publish events to event hubs, which are set up with either service endpoints or private endpoints. Regional virtual network integration is needed for your function app to connect to an event hub using a service endpoint or a private endpoint.

When you integrate Functions with a virtual network and enable `vnetRouteAllEnabled`, all outbound traffic from the function app is forced through the virtual network. This is particularly important for scenarios where you want to secure your function app by ensuring all traffic, including traffic to Azure services, goes through your virtual network for inspection and control. If you want to fully lock down your function app, you also need to [restrict your storage account](#).

To trigger (consume) events in a virtual network environment, the function app needs to be hosted in a Premium plan, a Dedicated (App Service) plan, or an App Service

Environment (ASE).

Additionally, running in an Azure Functions Premium plan and consuming events from a virtual network restricted Event Hub requires virtual network trigger support, also referred to as [runtime scale monitoring](#). Runtime scale monitoring can be configured via the Azure portal, Azure CLI, or other deployment solutions. Runtime scale monitoring isn't available when the function is running in a Dedicated (App Service) plan or an ASE.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [David Barkol](#) | Principal Solution Specialist GBB

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

Before continuing, consider reviewing these related articles:

- [Authorize access with Microsoft Entra ID](#)
- [Authorize access with a shared access signature in Azure Event Hubs](#)
- [Configure an identity-based resource](#)

Observability

Feedback

Was this page helpful?

 Yes

 No

Monitor Azure Functions and Event Hubs

Azure Event Hubs

Azure Functions

Azure Monitor

Monitoring provides insights into the behavior and health of your systems, and helps build a holistic view of the environment, historic trends, correlate diverse factors, and measure changes in performance, consumption, or error rate.

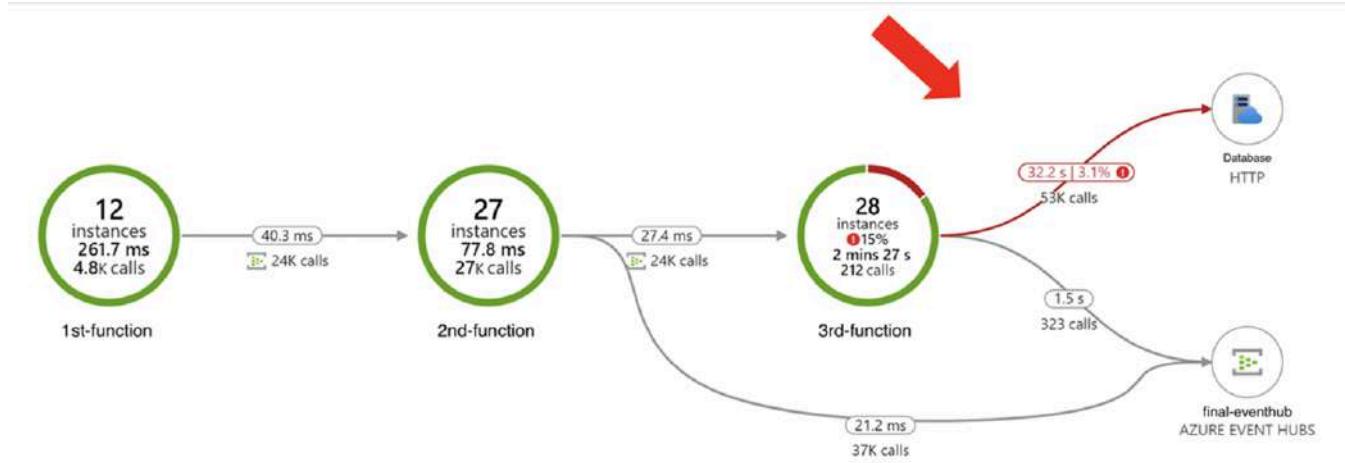
Azure Functions offers built-in integration with [Application Insights](#). From Application Insights, you can get information such as the number of function app instances or request and dependency telemetry of a function. When working with Functions and Event Hubs, Application Insights can also track the outgoing dependency telemetries to the event hub, calculating the processing time and showing the end-to-end flow of the system connected through Event Hubs.

This section introduces useful features and insights that you can get from Application Insights for your Event Hubs plus Functions solution.

Application Map

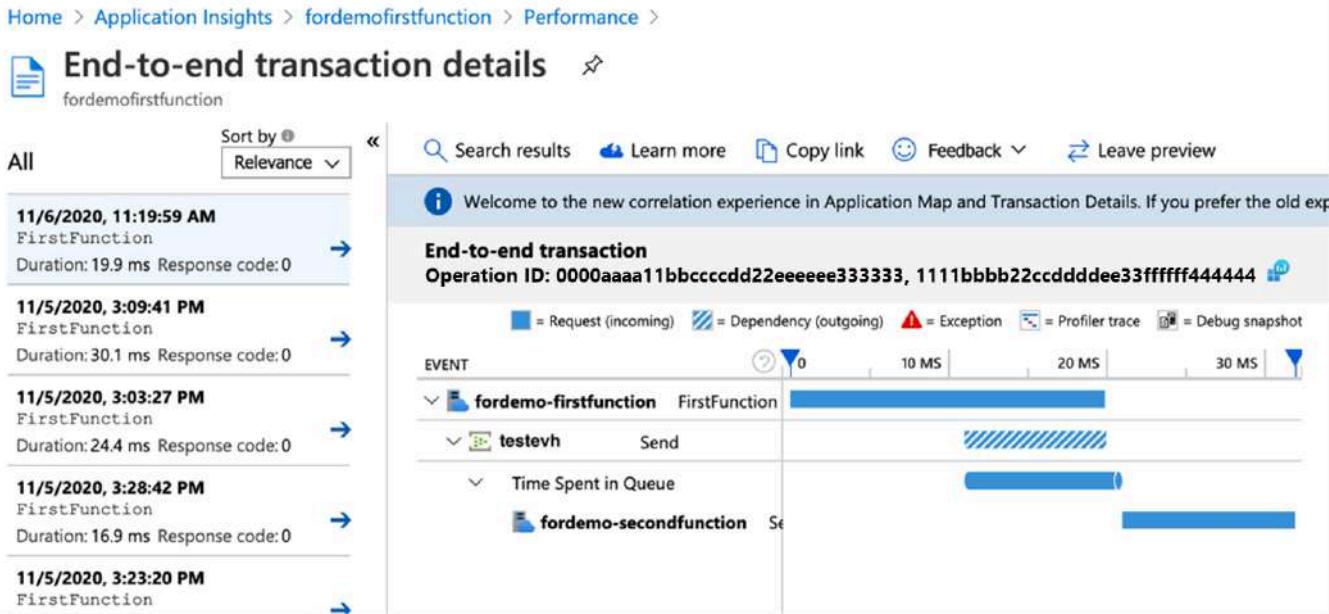
[Application Map](#) shows how the components in a system are interacting with each other. Because of the dependency telemetry that Application Insights provides, it maps out the flow of events between Azure Functions and Event Hubs, including the average of each function execution and average duration of an event in Event Hubs, as well as showing transactions that contain failures marked in red.

After sending the expected load to your system, you can go to Application Insights in the [Azure portal](#) ↗, and on the sidebar, choose **Application Map**. Here's a map that shows three functions, three event hubs, and apparent failures when writing to a downstream database:



End-to-end transaction details

End-to-end transaction details show how your system components interact with each other, in chronological order. This view also shows how long an event has spent in processing. You can also drill into the telemetry of each component from this view, which helps you troubleshoot across components within the same request when an issue occurred.



Platform metrics and telemetry

Platform-generated metrics in Azure Monitor for Event Hubs and Azure Functions can be used for overall monitoring of the solution behavior and health:

- [Azure Event Hubs metrics in Azure Monitor](#) are of interest to capture useful insights for Event Hubs (like aggregates of Incoming Requests, Outgoing Requests, Throttled Requests, Successful Requests, Incoming Messages, Outgoing Messages, Captured Messages, Incoming Bytes, Outgoing Bytes, Captured Bytes, User Errors).
- Azure Functions metrics share many of the metrics from [Azure App Service](#), with the addition of [Function Execution Count](#) and [Function Execution Units](#) that can be used for [understanding utilization and cost of the Consumption plan](#). Other metrics of interest are Connections, Data In, Data Out, Average Memory Working Set, Thread Count, Requests, and Response Time.

Azure Functions integrates with Application Insights to provide advanced and detailed telemetry and insights into the Functions host and function executions. To learn more, see [Analyze Azure Functions telemetry in Application Insights](#). When using Application Insights to monitor a topology, there are a variety of configurations available. To learn more, see [How to configure monitoring for Azure Functions](#).

The following is an example of extra telemetry for Event Hubs triggered functions generated in the **traces** table:

```
Trigger Details: PartitionId: 6, Offset: 3985758552064-3985758624640,  
EnqueueTimeUtc: 2022-10-31T12:51:58.1750000+00:00-2022-10-  
31T12:52:03.8160000+00:00, SequenceNumber: 3712266-3712275, Count: 10
```

This information requires that you use Event Hubs extension 4.2.0 or a later version. This data is very useful as it contains information about the message that triggered the function execution and can be used for querying and insights. It includes the following data for each time the function is triggered:

- The **partition ID** (6)
- The **partition offset** range (3985758552064-3985758624640)
- The **Enqueue Time** range in UTC (2022-10-31T12:51:58.1750000+00:00-2022-10-31T12:52:03.8160000+00:00)
- The **sequence number range** 3712266-3712275
- And the **count of messages** (10)

Refer to the [Example Application Insights queries](#) section for examples on how to use this telemetry.

Custom telemetry is also possible for different languages ([C# class library](#), [C# Isolated](#), [C# Script](#), [JavaScript](#), [Java](#), [PowerShell](#), and [Python](#)). This logging shows up in the **traces** table in Application Insights. You can create your own entries into Application Insights and add custom dimensions that can be used for querying data and creating custom dashboards.

Finally, when your function app connects to an event hub using an output binding, entries are also written to the [Application Insights Dependencies table](#).

1 dependencies

Completed. Showing results from the last 24 hours.

00:01.1 4,170 records

timestamp (UTC)	id	target	type	name	success	duration	performanceBucket
9/9/2021, 10:48:36.032 PM	f281e7b3f54e9...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	110.867	<250ms
9/9/2021, 10:48:36.032 PM	f281e7b3f54e9...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	110.867	<250ms
9/9/2021, 10:48:36.154 PM	030fbfe4cf55...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	18.28	<250ms
9/9/2021, 10:48:36.154 PM	030fbfe4cf55...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	18.28	<250ms
9/9/2021, 10:48:37.278 PM	b95ab2d6bdc...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	10.205	<250ms
9/9/2021, 10:48:37.278 PM	b95ab2d6bdc...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	10.205	<250ms
9/9/2021, 10:48:37.323 PM	931223fb2c633...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	20.138	<250ms
9/9/2021, 10:48:37.323 PM	931223fb2c633...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	20.138	<250ms
9/9/2021, 10:48:38.545 PM	d6f975f59d3a4...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	8.188	<250ms
9/9/2021, 10:48:38.545 PM	d6f975f59d3a4...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	8.188	<250ms
9/9/2021, 10:48:40.038 PM	089a7ad2cf8b8...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	8.935	<250ms
9/9/2021, 10:48:40.038 PM	089a7ad2cf8b8...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	8.935	<250ms
9/9/2021, 10:48:40.122 PM	141f7ae3a0dcc1...	sb://evhns-pk7qsyygv15uw-westus.servicebus.windows.net/evh...	Azure Event H...	Send	True	13.618	<250ms

For Event Hubs, the correlation is injected into the event payload, and you see a **Diagnostic-Id** property in events:

```
{
  "Body": "/your event,
  "Properties": {
    "Diagnostic-Id": "00-4d43caac0301954dbc76e5a786f6739a-6784fcb30fba654c-00",
  },
  "SystemProperties": {
    "x-opt-sequence-number": 72,
    "x-opt-offset": "4294975936",
    "x-opt-queued-time": "2020-11-04T16:46:46.955Z"
  }
}
```

This follows the [W3C Trace Context](#) format that's also used as **Operation Id** and **Operation Links** in telemetry created by Functions, which allows Application Insights to construct the correlation between event hub events and function executions, even when they're distributed.

When function processed a batch of events...

Operation Name	Telemetry type	Operation Id	Operation Links
FirstFunction	request	2295ea6e8de7aa4f87c49e711b60a1fc	
FirstFunction	trace	2295ea6e8de7aa4f87c49e711b60a1fc	
FirstFunction	dependency	2295ea6e8de7aa4f87c49e711b60a1fc	

Operation Name	Telemetry type	Operation Id	Operation Links
SecondFunction	request	d7d48c7d1af16c4eb129fe517a87608b	[{"operation_Id": "2295ea6e8de7aa4f87c49e711b60a1fc", "id": "ac0dc1c0f0a1a549"}, {"operation_Id": "2295ea6e8de7aa4f87c49e711b60a1fc", "id": "ac0dc1c0f0a1a549"}]
SecondFunction	trace	d7d48c7d1af16c4eb129fe517a87608b	

Example Application Insights queries

Below is a list of helpful Application Insights queries when monitoring Event Hubs with Azure Functions. This query display detailed information for event hub triggered function using **telemetry emitted by the Event Hubs extension 4.2.0 and greater**.

When [sampling is enabled](#) in Application Insights, there can be gaps in the data.

Detailed event processing information

The data is only emitted in the correct format when batched dispatch is used. Batch dispatch means that the function accepts multiple events for each execution, which is [recommended for performance](#). Keep in mind the following considerations:

- The `dispatchTimeMilliseconds` value approximates the length of time between when the event was written to the event hub and when it was picked up by the function app for processing.
- `dispatchTimeMilliseconds` can be negative or otherwise inaccurate because of clock drift between the event hub server and the function app.
- Event Hubs partitions are processed sequentially. A message won't be dispatched to function code for processing until all previous messages have been processed. Monitor the execution time of your functions as longer execution times will cause dispatch delays.
- The calculation uses the `enqueueTime` of the first message in the batch. Dispatch times might be lower for other messages in the batch.
- `dispatchTimeMilliseconds` is based on the point in time.

- Sequence numbers are per-partition, and duplicate processing can occur because Event Hubs does not guarantee exactly-once message delivery.

Kusto

```

traces
| where message startswith "Trigger Details: Parti"
| parse message with * "tionId: " partitionId:string , Offset: "
  offsetStart:string "-" offsetEnd:string", EnqueueTimeUtc: "
  enqueueTimeStart:datetime "+00:00-" enqueueTimeEnd:datetime "+00:00,
  SequenceNumber: "
  sequenceNumberStart:string "-" sequenceNumberEnd:string ", Count: "
  messageCount:int
| extend dispatchTimeMilliseconds = (timestamp - enqueueTimeStart) / 1ms
| project timestamp, cloud_RoleInstance, operation_Name, processId =
  customDimensions.ProcessId, partitionId, messageCount, sequenceNumberStart,
  sequenceNumberEnd, enqueueTimeStart, enqueueTimeEnd, dispatchTimeMilliseconds

```

Results Chart Columns Display time (UTC+00:00) Group columns

Completed. Showing results from the last 24 hours.

timestamp [UTC]	cloud_RoleInstance	operation_Name	processId	partitionId	messageCount	sequenceNumberStart
9/9/2021, 10:48:30.603 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	1	0
9/9/2021, 10:48:30.603 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	1	0
9/9/2021, 10:48:30.637 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	47	1
9/9/2021, 10:48:30.637 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	47	1
9/9/2021, 10:48:32.196 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	1	48
9/9/2021, 10:48:32.196 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	1	48
9/9/2021, 10:48:32.201 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	29	49
9/9/2021, 10:48:32.201 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	29	49
9/9/2021, 10:48:34.186 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	2	78
9/9/2021, 10:48:34.186 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	2	78
9/9/2021, 10:48:34.245 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	54	80
9/9/2021, 10:48:34.245 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	54	80
9/9/2021, 10:48:35.955 PM	aa11bb22cc33dd44ee55ff66aa77bb88cc99dd00bb22cc33dd44e...	ProcessorFunction	16592	0	1	134

Page 1 of 23 50 items per page 1 - 50 of 1140 items

Dispatch latency visualization

This query visualizes the 50th and 90th percentile event dispatch latency for a given event hub triggered function. For more information and notes, see the previous query.

Kusto

```

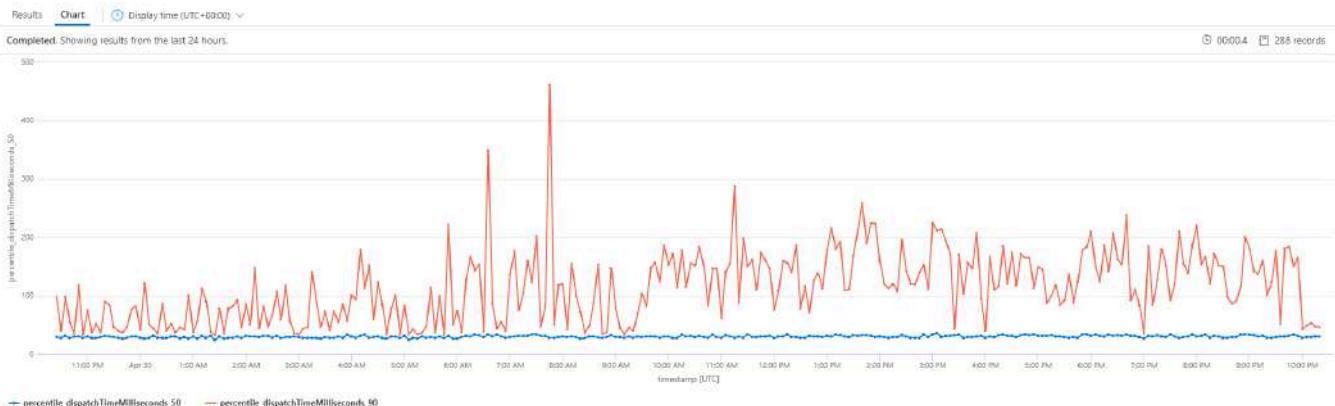
traces
| where operation_Name == "<ENTER THE NAME OF YOUR FUNCTION HERE>"
| where message startswith "Trigger Details: Parti"
| parse message with * "tionId: " partitionId:string , Offset: "
  offsetStart:string "-" offsetEnd:string", EnqueueTimeUtc: "
  enqueueTimeStart:datetime "+00:00-" enqueueTimeEnd:datetime "+00:00,
  SequenceNumber: "
  sequenceNumberStart:string "-" sequenceNumberEnd:string ", Count: "
  messageCount:int

```

```

| extend dispatchTimeMilliseconds = (timestamp - enqueueTimeStart) / 1ms
| summarize percentiles(dispatchTimeMilliseconds, 50, 90) by bin(timestamp, 5m)
| render timechart

```



Dispatch latency summary

This query is similar to above but shows a summary view.

Kusto

```

traces
| where message startswith "Trigger Details: Parti"
| parse message with * "tionId: " partitionId:string ", Offset: "
  offsetStart:string "-" offsetEnd:string", EnqueueTimeUtc: "
  enqueueTimeStart:datetime "+00:00-" enqueueTimeEnd:datetime "+00:00",
  SequenceNumber: "
  sequenceNumberStart:string "-" sequenceNumberEnd:string ", Count: "
  messageCount:int
| extend dispatchTimeMilliseconds = (timestamp - enqueueTimeStart) / 1ms
| summarize messageCount = sum(messageCount),
  percentiles(dispatchTimeMilliseconds, 50, 90, 99, 99.9, 99.99) by operation_Name

```

Results						
Completed. Showing results from the last 24 hours.						
operation_Name	messageCount	percentile_dispatchTimeMilliseconds_50	percentile_dispatchTimeMilliseconds_90	percentile_dispatchTimeMilliseconds_99	percentile_dispatchTimeMilliseconds_99.9	percentile_dispatchTimeMilliseconds_99.99
ProcessorFunction	88,572	47.417	1,954	15,484	17,925	17,925

Message distribution across partitions

This query shows how to visualize message distribution across partitions.

Kusto

```

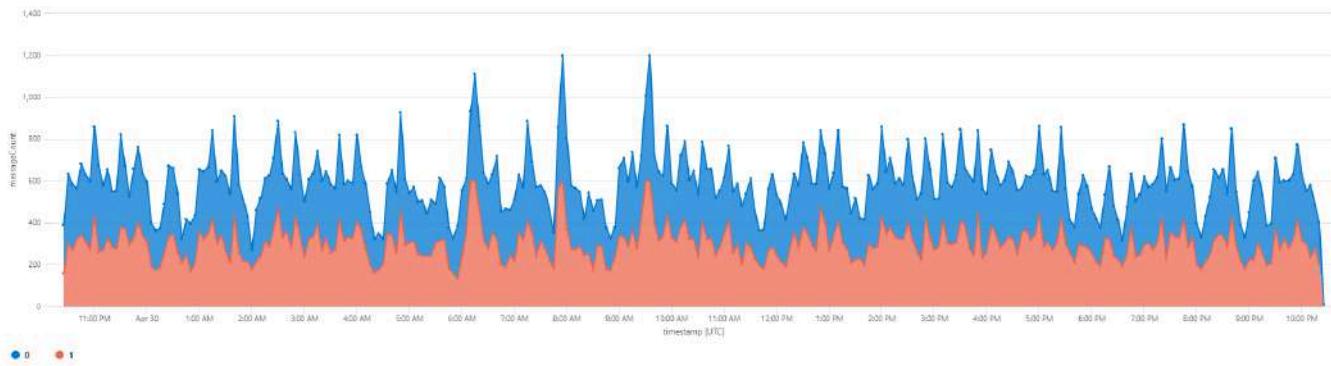
traces
| where message startswith "Trigger Details: Parti"
| parse message with * "tionId: " partitionId:string ", Offset: "
  offsetStart:string "-" offsetEnd:string", EnqueueTimeUtc: "

```

```

enqueueTimeStart:datetime "+00:00-" enqueueTimeEnd:datetime "+00:00",
SequenceNumber: "
sequenceNumberStart:string "-" sequenceNumberEnd:string ", Count: "
messageCount:int
| summarize messageCount = sum(messageCount) by cloud_RoleInstance,
bin(timestamp, 5m)
| render areachart kind=stacked

```



Message distribution across instances

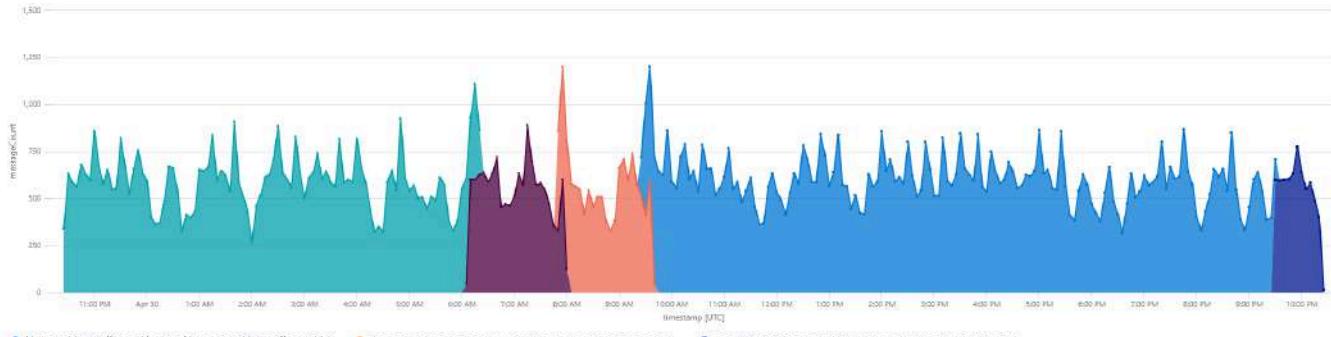
This query shows how to visualize message distribution across instances.

Kusto

```

traces
| where message startswith "Trigger Details: Parti"
| parse message with * "tionId: " partitionId:string , Offset: "
offsetStart:string "-" offsetEnd:string", EnqueueTimeUtc: "
enqueueTimeStart:datetime "+00:00-" enqueueTimeEnd:datetime "+00:00",
SequenceNumber: "
sequenceNumberStart:string "-" sequenceNumberEnd:string ", Count: "
messageCount:int
| summarize messageCount = sum(messageCount) by cloud_RoleInstance,
bin(timestamp, 5m)
| render areachart kind=stacked

```

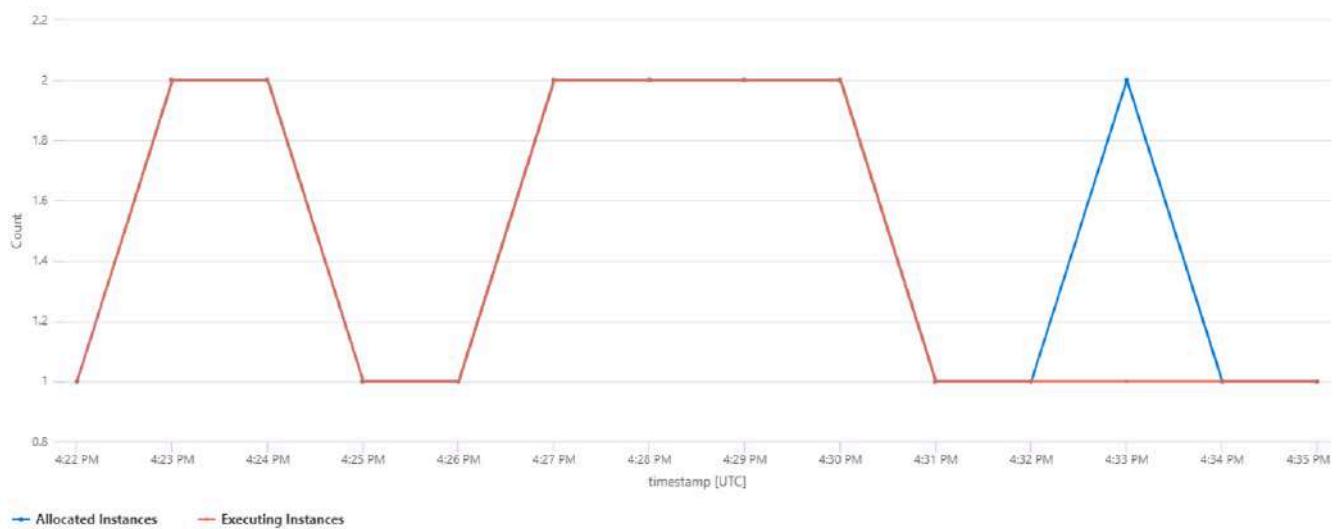


Executing Instances and Allocated Instances

This query shows how to visualize the number of Azure Functions instances that are processing events from Event Hubs, and the total number of instances (processing and waiting for lease). Most of the time they should be the same.

```
Kusto

traces
| where message startswith "Trigger Details: Parti"
| summarize type = "Executing Instances", Count = dcount(cloud_RoleInstance) by
bin(timestamp, 60s)
| union (
    traces
    | summarize type = "Allocated Instances", Count = dcount(cloud_RoleInstance)
by
bin(timestamp, 60s)
)
| project timestamp, type, Count
| render timechart
```



All Telemetry for a specific Function Execution

The `operation_Id` field can be used across the different tables in Application Insights. For Event Hubs triggered Azure Functions the following query for example will result in the trigger information, telemetry from logs inside the Function code, and dependencies and exceptions:

```
Kusto

union isfuzzy=true requests, exceptions, traces, dependencies
| where * has "<ENTER THE OPERATION_ID OF YOUR FUNCTION EXECUTION HERE>"
| order by timestamp asc
```

timestamp [UTC]	id	name	success	resultCode	duration	performanceBucket	itemType	customDimensions	operation_Name	operation_Id
9/9/2021, 10:48:30.588 PM	22cc22ccdd33e...	ProcessorFunction	True	0	40.67	<250ms	request	{"FunctionExecutionTimeMs":40.0062,"HostInstanceId":"52c5...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.588 PM	22cc22ccdd33e...	ProcessorFunction	True	0	40.67	<250ms	request	{"FunctionExecutionTimeMs":40.0062,"HostInstanceId":"52c5...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.603 PM							trace	{"prop__OriginalFormat":"Executing [functionName] (Reason:...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.603 PM							trace	{"prop__OriginalFormat":"Trigger Details: PartitionId: (PartitionId...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.603 PM							trace	{"prop__OriginalFormat":"Trigger Details: PartitionId: (PartitionId...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.603 PM							trace	{"prop__OriginalFormat":"Executing [functionName] (Reason:...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.612 PM							trace	{"prop__OriginalFormat":"DebatchingFunction: batchSize=1",...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.612 PM							trace	{"prop__OriginalFormat":"DebatchingFunction: batchSize=1",...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.620 PM							trace	{"prop__OriginalFormat":"No sensors matching yellow in this ...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.620 PM							trace	{"prop__OriginalFormat":"No sensors matching yellow in this ...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.620 PM							trace	{"prop__OriginalFormat":"DebatchingFunction: successfulMes...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.620 PM							trace	{"prop__OriginalFormat":"DebatchingFunction: successfulMes...	ProcessorFunction	aaaaaaaaaa00001111
9/9/2021, 10:48:30.624 PM							trace	{"prop__OriginalFormat":"TransformingFunction: successfulM...	ProcessorFunction	aaaaaaaaaa00001111

End-to-End Latency for an Event

As the `enqueueTimeUtc` property in the trigger detail trace shows the enqueue time of only the first event of each batch that the function processed, a more advanced query can be used to calculate the end-to-end latency of events between two functions with Event Hubs in between. This query will expand the operation links (if any) in the second function's request and map its end time to the same corresponding operation ID of the first function start time.

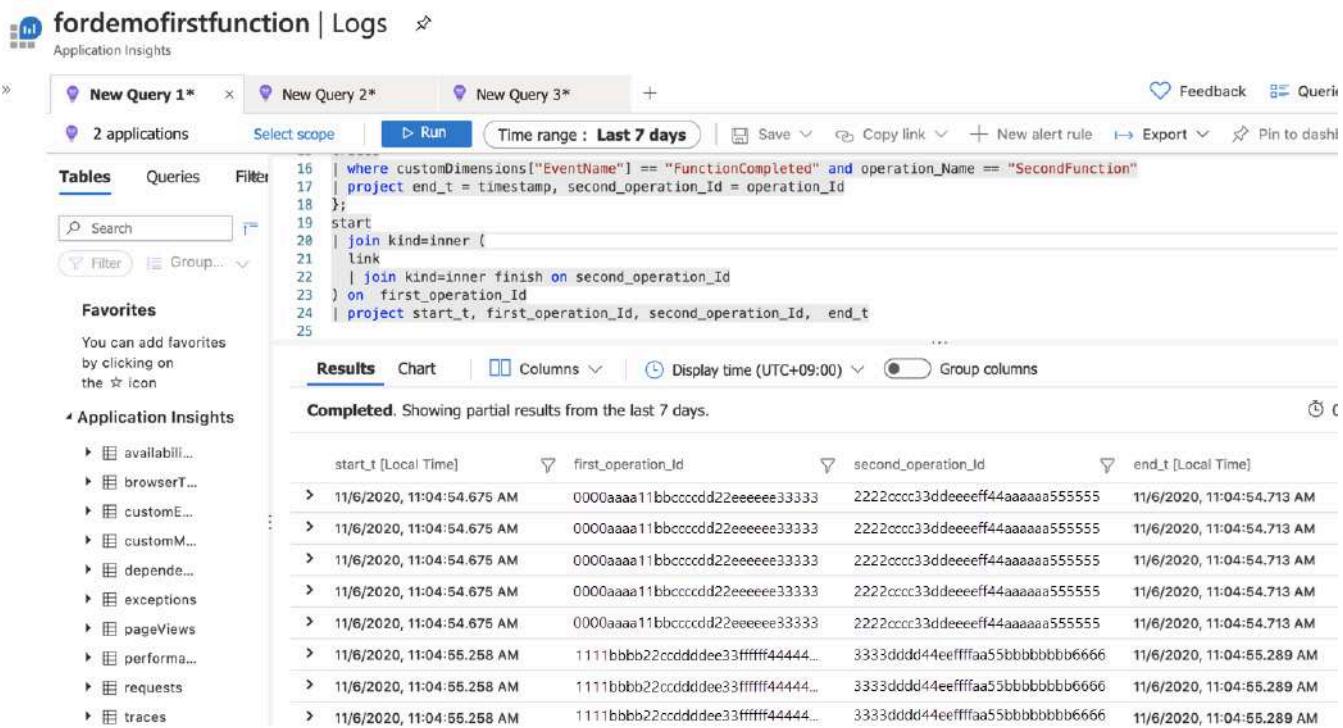
Kusto

```
let start = view(){
    requests
    | where operation_Name == "FirstFunction"
    | project start_t = timestamp, first_operation_Id = operation_Id
};

let link = view(){
    requests
    | where operation_Name == "SecondFunction"
    | mv-expand ex = parse_json(tostring(customDimensions["_MS.links"]))
    | extend parent = case(isnotempty(ex.operation_Id), ex.operation_Id, operation_Id)
    |
    | project first_operation_Id = parent, second_operation_Id = operation_Id
};

let finish = view(){
    traces
    | where customDimensions["EventName"] == "FunctionCompleted" and operation_Name
    == "SecondFunction"
    | project end_t = timestamp, second_operation_Id = operation_Id
};

start
| join kind=inner (
    link
    | join kind=inner finish on second_operation_Id
    ) on first_operation_Id
| project start_t, end_t, first_operation_Id, second_operation_Id
| summarize avg(datetime_diff('second', end_t, start_t))
```



The screenshot shows the Microsoft Azure Application Insights Logs interface. The top navigation bar includes 'fordemofirstfunction | Logs' and 'Application Insights'. Below the bar are tabs for 'New Query 1*', 'New Query 2*', and 'New Query 3*'. The 'Run' button is highlighted. The 'Time range' is set to 'Last 7 days'. The left sidebar has sections for 'Tables', 'Queries', 'Filter', 'Search', 'Favorites', and 'Application Insights'. Under 'Application Insights', there are links for 'availability', 'browserT...', 'customE...', 'customM...', 'depende...', 'exceptions', 'pageViews', 'performa...', 'requests', and 'traces'. The main area shows a Kusto query results table with columns: 'start_t [Local Time]', 'first_operation_Id', 'second_operation_Id', and 'end_t [Local Time]'. The table contains 10 rows of data, each with a timestamp from '11/6/2020, 11:04:54.675 AM' to '11/6/2020, 11:04:55.289 AM' and unique identifiers for operation IDs.

start_t [Local Time]	first_operation_Id	second_operation_Id	end_t [Local Time]
11/6/2020, 11:04:54.675 AM	0000aaaa11bbcccd22eeeeee33333	2222cccc33ddeeff44aaaaaa555555	11/6/2020, 11:04:54.713 AM
11/6/2020, 11:04:54.675 AM	0000aaaa11bbcccd22eeeeee33333	2222cccc33ddeeff44aaaaaa555555	11/6/2020, 11:04:54.713 AM
11/6/2020, 11:04:54.675 AM	0000aaaa11bbcccd22eeeeee33333	2222cccc33ddeeff44aaaaaa555555	11/6/2020, 11:04:54.713 AM
11/6/2020, 11:04:54.675 AM	0000aaaa11bbcccd22eeeeee33333	2222cccc33ddeeff44aaaaaa555555	11/6/2020, 11:04:54.713 AM
11/6/2020, 11:04:54.675 AM	0000aaaa11bbcccd22eeeeee33333	2222cccc33ddeeff44aaaaaa555555	11/6/2020, 11:04:54.713 AM
11/6/2020, 11:04:55.258 AM	1111bbbb22cddddee33fffff44444...	3333dddd44eeffffaa55bbbbbbb6666	11/6/2020, 11:04:55.289 AM
11/6/2020, 11:04:55.258 AM	1111bbbb22cddddee33fffff44444...	3333dddd44eeffffaa55bbbbbbb6666	11/6/2020, 11:04:55.289 AM
11/6/2020, 11:04:55.258 AM	1111bbbb22cddddee33fffff44444...	3333dddd44eeffffaa55bbbbbbb6666	11/6/2020, 11:04:55.289 AM

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [David Barkol](#) | Principal Solution Specialist GBB

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

To learn more, consider reviewing these related articles:

- [Analyze Azure Functions telemetry in Application Insights](#)
- [Configure monitoring for Azure Functions](#)
- [Analyze Azure Functions telemetry in Application Insights](#)
- [Metrics in Azure Monitor - Azure Event Hubs](#)
- [Kusto Query Language](#)

Apache Kafka migration to Azure

Azure HDInsight

Azure Cosmos DB

Azure Data Lake Storage

Azure Stream Analytics

Apache Kafka [↗](#) is a highly scalable and fault tolerant distributed messaging system that implements a publish-subscribe architecture. It's used as an ingestion layer in real-time streaming scenarios, such as Internet of Things and real-time log monitoring systems. It's also used increasingly as the immutable append-only data store in Kappa architectures.

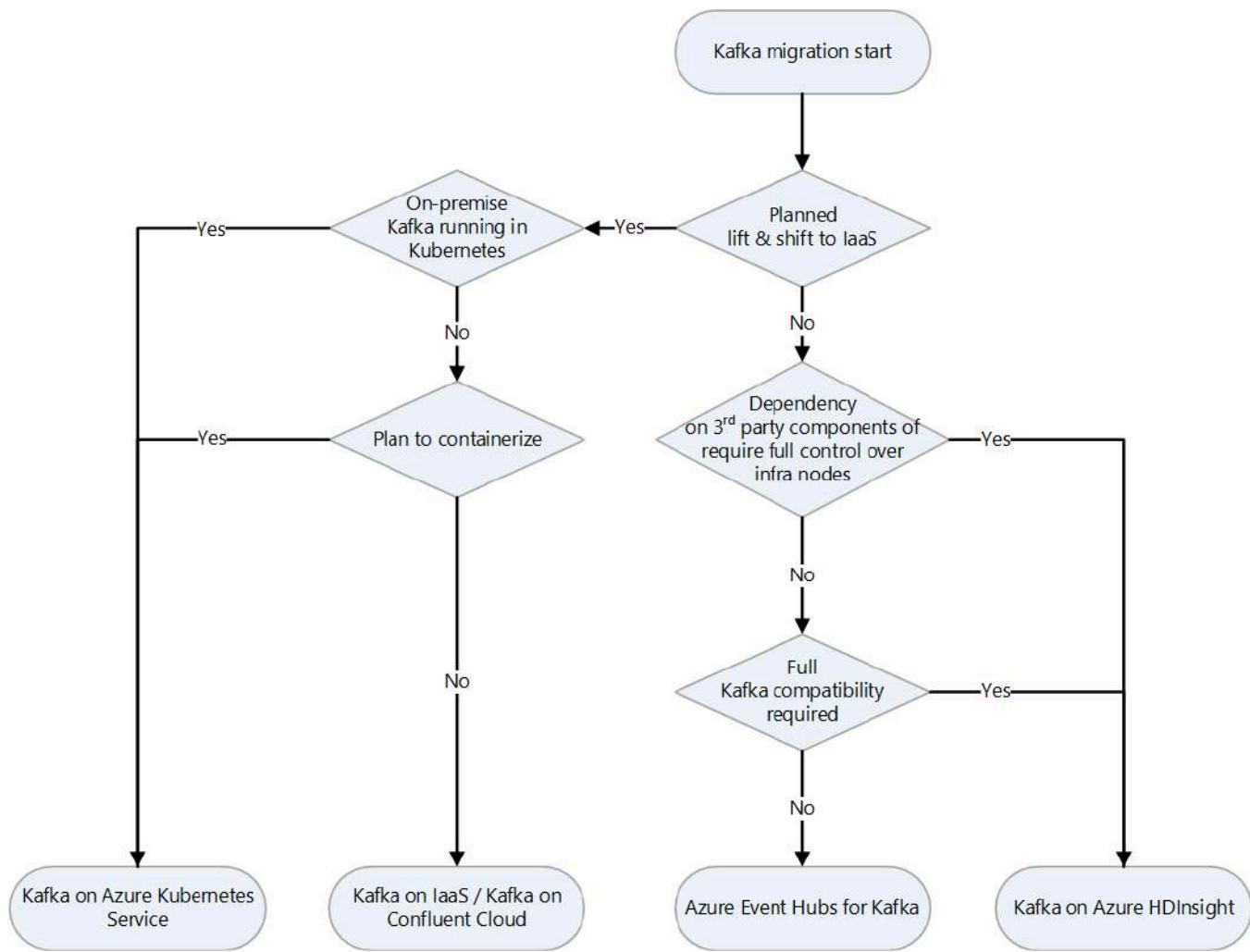
Apache [↗](#)®, Apache Spark® [↗](#), Apache Hadoop® [↗](#), Apache HBase [↗](#), Apache Storm® [↗](#), Apache Sqoop® [↗](#), Apache Kafka® [↗](#), and the flame logo are either registered trademarks or trademarks of the Apache Software Foundation in the United States and/or other countries. No endorsement by The Apache Software Foundation is implied by the use of these marks.

Migration approach

This article presents various strategies for migrating Kafka to Azure:

- [Migrate Kafka to Azure infrastructure as a service \(IaaS\)](#)
- [Migrate Kafka to Azure Event Hubs for Kafka](#)
- [Migrate Kafka on Azure HDInsight](#)
- [Use Azure Kubernetes Service \(AKS\) with Kafka on HDInsight](#)
- [Use Kafka on AKS with the Strimzi Operator](#)

Here's a decision flowchart for deciding which strategy to use.



Migrate Kafka to Azure IaaS

For one way to migrate Kafka to Azure IaaS, see [Kafka on Ubuntu virtual machines](#).

Migrate Kafka to Event Hubs for Kafka

Event Hubs provides an endpoint that's compatible with the Apache Kafka producer and consumer APIs. Most Apache Kafka client applications can use this endpoint, so you can use it as an alternative to running a Kafka cluster on Azure. The endpoint supports clients that use API versions 1.0 and later. For more information about this feature, see [Event Hubs for Apache Kafka overview](#).

To learn how to migrate your Apache Kafka applications to use Event Hubs, see [Migrate to Event Hubs for Apache Kafka ecosystems](#).

Features of Kafka and Event Hubs

[Expand table](#)

Similarities between Kafka and Event Hubs	Differences in Kafka and Event Hubs
Use partitions	Platform as a service versus software
Partitions are independent	Partitioning
Use a client-side cursor concept	APIs
Can scale to very high workloads	Runtime
Nearly identical conceptually	Protocols
Neither uses the HTTP protocol for receiving	Durability
	Security
	Throttling

Partitioning differences

 [Expand table](#)

Kafka	Event Hubs
Partition count manages scale.	Throughput units manage scale.
You must load balance partitions across machines.	Load balancing is automatic.
You must manually reshuffle by using split and merge.	Repartitioning isn't required.

Durability differences

 [Expand table](#)

Kafka	Event Hubs
Volatile by default	Always durable
Replicated after an acknowledgment (ACK) is received	Replicated before an ACK is sent
Depends on disk and quorum	Provided by storage

Security differences

 [Expand table](#)

Kafka	Event Hubs
Secure Sockets Layer (SSL) and Simple Authentication and Security Layer (SASL)	Shared Access Signature (SAS) and SASL or PLAIN RFC 4618
File-like access control lists	Policy
Optional transport encryption	Mandatory Transport Layer Security (TLS)
User based	Token based (unlimited)

Other differences

 [Expand table](#)

Kafka	Event Hubs
Doesn't throttle	Supports throttling
Uses a proprietary protocol	Uses AMQP 1.0 protocol
Doesn't use HTTP for send	Uses HTTP send and batch send

Migrate Kafka on HDInsight

You can migrate Kafka to Kafka on HDInsight. For more information, see [What is Apache Kafka in HDInsight?](#)

Use AKS with Kafka on HDInsight

For more information, see [Use AKS with Apache Kafka on HDInsight](#).

Use Kafka on AKS with the Strimzi Operator

For more information, see [Deploy a Kafka cluster on AKS by using Strimzi](#).

Kafka data migration

You can use Kafka's [MirrorMaker tool](#) to replicate topics from one cluster to another. This technique can help you migrate data after a Kafka cluster is provisioned. For more information, see [Use MirrorMaker to replicate Apache Kafka topics with Kafka on HDInsight](#).

The following migration approach uses mirroring:

1. Move producers first. When you migrate the producers, you prevent production of new messages on the source Kafka.
2. After the source Kafka consumes all remaining messages, you can migrate the consumers.

The implementation includes the following steps:

1. Change the Kafka connection address of the producer client to point to the new Kafka instance.
2. Restart the producer business services and send new messages to the new Kafka instance.
3. Wait for the data in the source Kafka to be consumed.
4. Change the Kafka connection address of the consumer client to point to the new Kafka instance.
5. Restart the consumer business services to consume messages from the new Kafka instance.
6. Verify that consumers succeed in getting data from the new Kafka instance.

Monitor the Kafka cluster

You can use Azure Monitor logs to analyze logs that Apache Kafka on HDInsight generates. For more information, see [Analyze logs for Apache Kafka on HDInsight](#).

Apache Kafka Streams API

The Kafka Streams API makes it possible to process data in near real-time and to join and aggregate data. For more information, see [Introducing Kafka Streams: Stream Processing Made Simple - Confluent](#).

The Microsoft and Confluent partnership

Confluent provides a cloud-native service for Apache Kafka. Microsoft and Confluent have a strategic alliance. For more information, see the following resources:

- [Confluent and Microsoft announce strategic alliance](#)
- [Introducing seamless integration between Microsoft Azure and Confluent Cloud](#)

Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal authors:

- [Namrata Maheshwary](#) | Senior Cloud Solution Architect
- [Raja N](#) | Director, Customer Success
- [Hideo Takagi](#) | Cloud Solution Architect
- [Ram Yerrabotu](#) | Senior Cloud Solution Architect

Other contributors:

- [Ram Baskaran](#) | Senior Cloud Solution Architect
- [Jason Bouska](#) | Senior Software Engineer - Azure Patterns & Practices
- [Eugene Chung](#) | Senior Cloud Solution Architect
- [Pawan Hosattu](#) | Senior Cloud Solution Architect - Engineering
- [Daman Kaur](#) | Cloud Solution Architect
- [Danny Liu](#) | Senior Cloud Solution Architect - Engineering
- [Jose Mendez](#) | Senior Cloud Solution Architect
- [Ben Sadeghi](#) | Senior Specialist
- [Sunil Sattiraju](#) | Senior Cloud Solution Architect
- [Amanjeet Singh](#) | Principal Program Manager
- [Nagaraj Seeplapudur Venkatesan](#) | Senior Cloud Solution Architect - Engineering

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

Azure product introductions

- [Introduction to Azure Data Lake Storage](#)
- [What is Apache Spark in HDInsight?](#)
- [What is Apache Hadoop in HDInsight?](#)
- [What is Apache HBase in HDInsight?](#)
- [What is Apache Kafka in HDInsight?](#)
- [Overview of enterprise security in HDInsight](#)

Azure product reference

- [Microsoft Entra documentation](#)
- [Azure Cosmos DB documentation](#)
- [Azure Data Factory documentation](#)
- [Azure Databricks documentation](#)
- [Event Hubs documentation](#)
- [Azure Functions documentation](#)
- [HDInsight documentation](#)
- [Microsoft Purview data governance documentation](#)
- [Azure Stream Analytics documentation](#)

Other

- [Enterprise Security package for HDInsight](#)
- [Develop Java MapReduce programs for Apache Hadoop on HDInsight](#)
- [Use Apache Sqoop with Hadoop in HDInsight](#)
- [Overview of Apache Spark Streaming](#)
- [Structured Streaming tutorial](#)
- [Use Event Hubs from Apache Kafka applications](#)

Azure API Management landing zone architecture

Azure API Management

Azure Application Gateway

Azure Functions

.NET

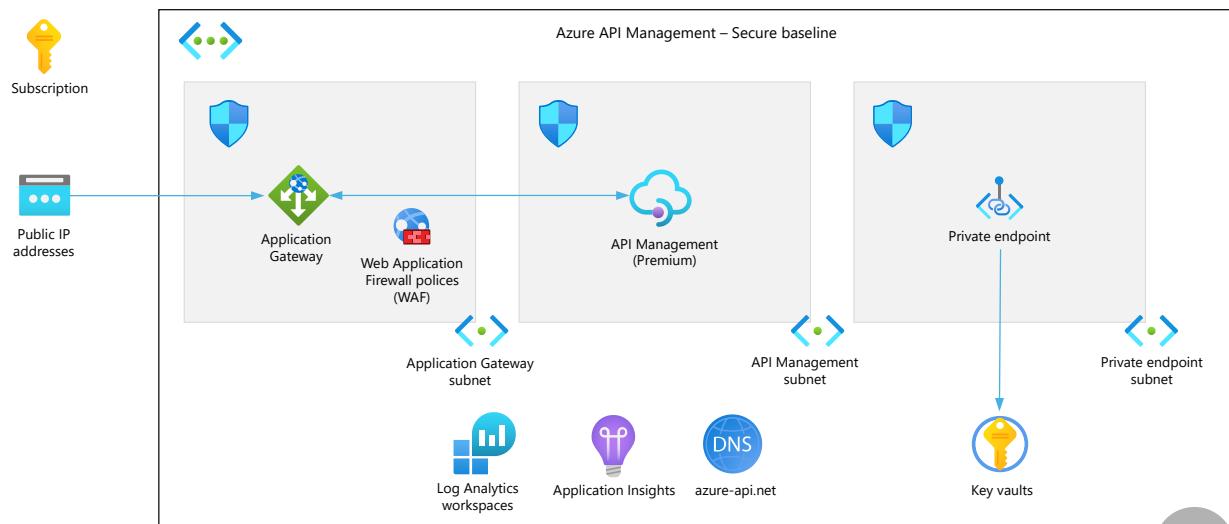
APIs have become increasingly integral to how organizations and their customers access services, both within internal systems and through external channels. Internally, APIs facilitate access to line-of-business (LoB) applications, proprietary solutions, and partner integrations. Externally, a growing number of organizations are focused on enhancing productivity and generating revenue through API monetization. Given this trend, Azure API Management serves as a foundational element in the standardized governance, publication, and oversight of APIs for both internal and external stakeholders.

Azure Application Gateway serves as a security checkpoint for APIs. Instead of allowing users to connect directly over the internet, you route all traffic through an application gateway. This setup adds extra access controls to help protect your APIs. With this approach, you can use a single API Management instance to support both internal APIs within your organization and external APIs outside your organization, while keeping any publicly exposed APIs secured behind the gateway.

! Note

This architecture serves as the foundation of the guidance for [API Management in an Azure landing zone](#) in the Cloud Adoption Framework for Azure.

Architecture



Download a [Visio file](#) of this architecture.

This architecture assumes that the policies are in place from the [Azure landing zone reference implementation](#) and that the structure is driven downward from the management group.

Workflow

- Public IP addresses are assigned to an application gateway, which serves as the entry point for external traffic. That endpoint exposes APIs through a custom domain.
- The application gateway is deployed in its own subnet and protected by Web Application Firewall (WAF) policies to inspect and filter incoming requests.
- Traffic is routed from the application gateway to API Management (Premium), which resides in a separate API Management subnet. The API Management instance is configured in internal mode, which prevents direct public access.
- Private endpoints are used to securely connect API Management to back-end application servers that are exposed only to the virtual network. API Management also periodically connects dependencies, such as Azure key vaults. Typically, all of this private connectivity occurs with endpoints in a dedicated private endpoint subnet.
- Log Analytics workspaces and Application Insights are integrated for logging, monitoring, and telemetry.

Components

- [API Management](#) is a managed service that allows you to manage services across hybrid and multicloud environments. It provides control and security for API observability and consumption by both internal and external users. In this architecture, API Management serves as a facade to abstract the back-end architecture.
- [Application Gateway](#) is a managed service that serves as a layer-7 load balancer and [WAF](#). Application Gateway protects the internal API Management instance, which enables the use of both internal and external modes. In this architecture, API Management secures APIs, and Application Gateway adds complementary capabilities such as WAF.
- [Private Domain Name System \(DNS\) zones](#) are a feature of Azure DNS that allows you to manage and resolve domain names within a virtual network without needing to implement a custom DNS solution. A private DNS zone can be aligned to one or more virtual networks through [virtual network links](#). In this architecture, a private DNS zone is required to ensure proper name resolution within the virtual network.

- [Application Insights](#) is an extensible application performance management service that helps developers detect anomalies, diagnose problems, and understand usage patterns. Application Insights features extensible application performance management and monitoring for live web apps. Various platforms are supported, including .NET, Node.js, Java, and Python. It supports apps that are hosted in Azure, on-premises, in a hybrid environment, or in other public clouds. In this architecture, Application Insights monitors the behaviors of the deployed application.
- [Log Analytics](#) is a cloud-based data analysis tool that enables you to edit and run log queries against data in Azure Monitor Logs, optionally from within the Azure portal. Developers can run simple queries to retrieve records or use Log Analytics for advanced analysis, then visualize the results. In this architecture, Log Analytics aggregates all the platform resource logs for analysis and reporting.
- [Azure Key Vault](#) is a cloud service that securely stores and accesses secrets. These secrets range from API keys and passwords to certificates and cryptographic keys. In this architecture, Key Vault stores the Secure Sockets Layer (SSL) certificates that Application Gateway uses.

Alternatives

For the back-end services that the API Management instance connects to, several alternatives are available:

- [Azure App Service](#) is a fully managed HTTP-based service that builds, deploys, and scales web apps. It supports .NET, .NET Core, Java, Ruby, Node.js, PHP, and Python. Applications can run and scale in either Windows or Linux-based environments.
- [Azure Kubernetes Service \(AKS\)](#) is a managed Kubernetes offering that delivers fully managed clusters. It enables integrated continuous integration and continuous delivery (CI/CD), along with built-in governance and security.
- [Azure Logic Apps](#) is a cloud-based platform that creates and runs automated workflows. For more information, see an [example reference architecture](#).
- [Azure Container Apps](#) is a fully managed serverless container service that enables you to run microservices and containerized applications on a serverless platform.

For multiregion deployments, consider using [Azure Front Door](#) to provide fast, reliable, and secure access between your users and your applications' static and dynamic web content.

To see additional examples of how Application Gateway can protect APIs, see [Protect APIs with Application Gateway and API Management](#).

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

Reliability

Reliability helps ensure that your application can meet the commitments that you make to your customers. For more information, see [Design review checklist for Reliability](#).

- Deploy at least two [scale units](#) of API Management that are spread over two or more availability zones in each region. [Monitor the capacity metrics](#), and provision sufficient capacity units so that you can continue to operate even if the units in one availability zone are lost.
- We recommend that you use the Premium tier because it supports availability zones and multiregion deployments. This capability means that your services can continue to run even if one region or zone goes down. These features help protect your application during outages or disasters.
- For disaster recovery, set up API Management with a user-assigned managed identity instead of a system-assigned identity. If you redeploy or delete the resource, the identity and its permissions remain in place, so you can restore access more easily. Use Azure Pipelines to automate backups. Decide if you need to deploy your services in more than one region for better reliability.
- Virtual network peering provides strong performance within a region, but it has a scalability limit of 500 networks. If you need to connect more workloads, use a [hub-spoke design](#) or [Azure Virtual WAN](#).

Security

Security provides assurances against deliberate attacks and the misuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

- API Management [validation policies](#) validate API requests and responses against an OpenAPI schema. These features aren't a replacement for a [WAF](#), but they can provide extra protection against some threats. Adding validation policies can have performance implications, so we recommend that you use performance load tests to assess their impact on API throughput.

- Microsoft Defender for APIs provides full life cycle protection, detection, and response for APIs published in API Management. One key capability is detecting exploits of the Open Web Application Security Project (OWASP) API Top 10 vulnerabilities through runtime anomaly observations by using machine learning-based and rule-based detections.
- API Management [workspaces](#) help you organize and isolate your APIs. This approach makes it easier to control who can access and manage them. Each workspace can have its own set of permissions, so you can limit access to only the people or teams who need it. This separation reduces the risk of accidental changes or unauthorized access and supports a more secure API environment.
- Use [Key Vault secrets as named values in API Management policies](#) to protect sensitive information in API Management policies.
- Use [Application Gateway for external access of an internal API Management instance](#) to protect the API Management instance, defend against common web application exploits and vulnerabilities by using WAF, and enable hybrid connectivity.
- Deploy the API Management gateway in a virtual network to support hybrid connectivity and increased security.
- Virtual network peering improves performance in a region and enables private communication between virtual networks.
- When you use a WAF, you introduce a layer that inspects incoming traffic for malicious behavior. This protection helps block common threats such as SQL injection and cross-site scripting. Application Gateway and distributed denial-of-service (DDoS) protection help prevent excessive traffic or connection floods from overwhelming the API Management instance. For more information, see [Protect APIs by using Application Gateway and API Management](#).
- Private endpoints for Azure Functions allow you to securely connect to your function apps over a private IP address within your virtual network. This setup prevents exposure of your functions to the public internet, which reduces the risk of unauthorized access. In this architecture, private endpoints ensure that only trusted resources within your network can access Azure Functions.

Cost Optimization

Cost Optimization focuses on ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

- This deployment uses the [Premium plan](#) to support availability zone and virtual network capabilities. If you don't require dedicated instances, you can also use [Flex Consumption](#), which supports both network access and availability zones. Review the [pricing calculator](#) for this deployment.
- For proof of concepts or prototypes, we recommend that you use other API Management tiers, such as Developer or Standard.

Operational Excellence

Operational Excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Design review checklist for Operational Excellence](#).

- Represent API Management configurations as Azure Resource Manager templates, and adopt an infrastructure as code (IaC) approach.
- Use a CI/CD process to manage, version, and update API Management configurations.
- Create custom health probes to help validate the status of your API Management instance. Use the URL `/status-0123456789abcdef` to create a common health endpoint for the API Management service in the application gateway.
- Certificates updated in the key vault are automatically rotated in API Management, which reflects the changes within four hours.
- If you use a DevOps tool, such as Azure DevOps or GitHub, then cloud-hosted agents or runners operate over the public internet. Because API Management in this architecture is set to an internal network, you need to use a DevOps agent that has access to the virtual network. The DevOps agent helps you deploy policies and other changes to the APIs in your architecture. You can use these [CI/CD templates](#) to separate the process into parts so that your development teams can deploy changes for each API. DevOps runners initiate the templates to handle these individual deployments.

Deploy this scenario

This architecture is available on [GitHub](#). It contains all the necessary IaC files and the [deployment instructions](#).

Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal authors:

- [Pete Messina](#) | Senior Cloud Solution Architect
- [Anthony Nevico](#) | Senior Cloud Solution Architect

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

- [Cloud Adoption Framework guidance for adopting API Management in an Azure landing zone](#)
- [API Management terminology](#)
- [Application Gateway documentation](#)

Related resources

- [Use API gateways in microservices](#)
- [Hub-spoke network topology in Azure](#)
- [Basic enterprise integration on Azure](#)
- [Protect APIs by using Application Gateway and API Management](#)

Basic enterprise integration on Azure

Microsoft Entra ID

Azure API Management

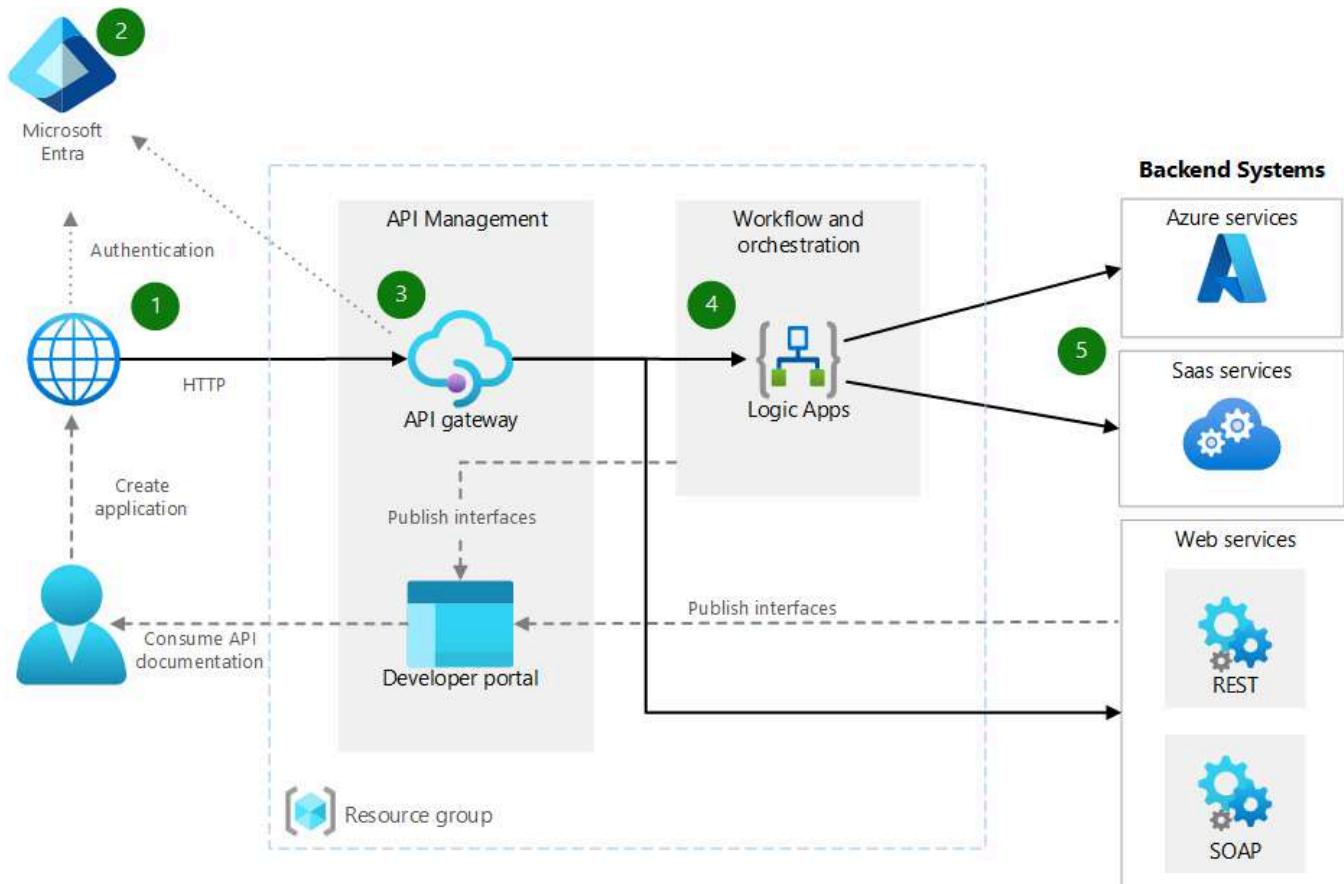
Azure DNS

Azure Logic Apps

Azure Monitor

This reference architecture uses [Azure Integration Services](#) to orchestrate calls to enterprise backend systems. The backend systems can include software as a service (SaaS) systems, Azure services, and existing web services in your enterprise.

Architecture



Download a [Visio file](#) of this architecture.

Workflow

- 1. Application.** The application is a client that calls the API gateway after authenticating with Microsoft Entra. The application can be a web app, mobile app, or any other client that can make HTTP requests.

2. **Microsoft Entra ID.** Is used to authenticate the client application. The client application obtains an access token from Microsoft Entra ID and includes it in the request to the API gateway.

3. **Azure API Management.** API Management consists of two related components:

- **API gateway.** The API gateway accepts HTTP call from the client application, validates the token from Microsoft Entra ID, and forwards the request to the backend service. The API gateway can also transform requests and responses, and cache responses.
- **Developer portal.** The [developer portal](#) is used by developers to discover and interact with the APIs. The developer portal can be customized to match your organization's branding.

4. **Azure Logic Apps.** Logic apps are used to orchestrate the calls to the backend services.

Logic apps can be triggered by a variety of events and can call a variety of services. In this solution, Logic Apps is used to call the backend services and provide connectivity through [connectors](#), reducing the need for custom code.

5. **Backend services.** The backend services can be any service or line of business application, such as a database, a web service, or a SaaS application. The backend services can be hosted in Azure or on-premises.

Components

- [Integration Services](#) is a collection of services that you can use to integrate applications, data, and processes. In this solution, two of these services are used: Logic Apps and API Management. Logic Apps is used to facilitate message based integration between systems and facilitate connectivity with connectors. API Management is used to provide a façade for the backend services, allowing for a consistent interface for clients to interact with.
 - [Logic Apps](#) is a serverless platform for building enterprise workflows that integrate applications, data, and services. In this solution Logic Apps is used to orchestrate the calls to the backend services and provide connectivity through connectors, reducing the need for custom code.
 - [API Management](#) is a managed service for publishing catalogs of HTTP APIs. You can use it to promote the reuse and discoverability of your APIs and to deploy an API gateway to proxy API requests. In this solution, API Management provides additional capabilities such as rate limiting, authentication, and caching to the backend services. Additionally, API Management provides a developer portal for clients to discover and interact with the APIs.

- [Azure DNS](#) is a hosting service for DNS domains. Azure DNS is hosting the public DNS records for the API Management service. This allows clients to resolve the DNS name to the IP address of the API Management service.
- [Microsoft Entra ID](#) is a cloud-based identity and access management service. Enterprise employees can use Microsoft Entra ID to access external and internal resources. Here Entra ID is used to secure the API Management service using [OAuth 2.0](#) and the developer portal using [Microsoft Entra External ID](#).

Scenario details

Integration Services is a collection of services that you can use to integrate applications, data, and processes for your enterprise. This architecture uses two of those services: [Logic Apps](#) to orchestrate workflows, and [API Management](#) to create catalogs of APIs.

In this architecture, composite APIs are built by [importing logic apps](#) as APIs. You can also import existing web services by [importing OpenAPI](#) (Swagger) specifications or [importing SOAP APIs](#) from WSDL specifications.

The API gateway helps to decouple front-end clients from the back end. For example, it can rewrite URLs, or transform requests before they reach the backend. It also handles many cross-cutting concerns such as authentication, cross-origin resource sharing (CORS) support, and response caching.

Potential use cases

This architecture is sufficient for basic integration scenarios in which the workflow is triggered by synchronous calls to backend services. A more sophisticated architecture using [queues and events](#) builds on this basic architecture.

Recommendations

Your specific requirements might differ from the generic architecture shown here. Use the recommendations in this section as a starting point.

API Management

Use the API Management Basic, Standard, or Premium tiers. These tiers offer a production service level agreement (SLA) and support scale-out within the Azure region. Throughput capacity for API Management is measured in *units*. Each pricing tier has a maximum scale-out. The Premium tier also supports scale-out across multiple Azure regions. Choose your tier

based on your feature set and the level of required throughput. For more information, see [API Management pricing](#) and [Capacity of an Azure API Management instance](#).

The API Management Consumption tier is not recommended for this solution because it doesn't support the developer portal which is required for this architecture. The Developer Tier is specifically for non-production environments and is not recommended for production workloads. A feature matrix detailing the differences between the tiers can be found [here](#).

Each Azure API Management instance has a default domain name, which is a subdomain of `azure-api.net`, for example, `contoso.azure-api.net`. Consider configuring a [custom domain](#) for your organization.

Logic Apps

Logic Apps works best in scenarios that don't require low latency for a response, such as asynchronous or semi long-running API calls. If low latency is required, for example in a call that blocks a user interface, use a different technology. For example, use Azure Functions or a web API deployed to Azure App Service. Use API Management to front the API to your API consumers.

Region

To minimize network latency, put API Management and Logic Apps in the same region. In general, choose the region that's closest to your users (or closest to your backend services).

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Microsoft Azure Well-Architected Framework](#).

Reliability

Reliability ensures your application can meet the commitments you make to your customers. For more information, see [Design review checklist for Reliability](#).

Review the SLAs for each service [here](#)

If you deploy API Management across two or more regions with Premium tier, it's eligible for a higher SLA. See [API Management pricing](#).

Backups

Regularly [back up](#) your API Management configuration. Store your backup files in a location or Azure region that differs from the region where the service is deployed. Based on your RTO, choose a disaster recovery strategy:

- In a disaster recovery event, provision a new API Management instance, restore the backup to the new instance, and repoint the DNS records.
- Keep a passive instance of the API Management service in another Azure region. Regularly restore backups to that instance, to keep it in sync with the active service. To restore the service during a disaster recovery event, you need only repoint the DNS records. This approach incurs additional costs because you pay for the passive instance, but it reduces recovery time.

For logic apps, we recommend a configuration-as-code approach to backing up and restoring. Because logic apps are serverless, you can quickly recreate them from Azure Resource Manager templates. Save the templates in source control, integrate the templates with your continuous integration/continuous deployment (CI/CD) process. In a disaster recovery event, deploy the template to a new region.

If you deploy a logic app to a different region, update the configuration in API Management. You can update the API's **Backend** property by using a basic PowerShell script.

Security

Security provides assurances against deliberate attacks and the abuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

Although this list doesn't completely describe all security best practices, here are some security considerations that apply specifically to this architecture:

- The Azure API Management service has a fixed public IP address. Restrict access for calling Logic Apps endpoints to only the IP address of API Management. For more information, see [Restrict inbound IP addresses](#).
- To make sure users have appropriate access levels, use Azure role-based access control (Azure RBAC).
- Secure public API endpoints in API Management by using OAuth or OpenID Connect. To secure public API endpoints, configure an identity provider, and add a JSON Web Token (JWT) validation policy. For more information, see [Protect an API by using OAuth 2.0 with Microsoft Entra ID and API Management](#).

- Connect to back-end services from API Management by using [mutual certificates](#).
- Enforce HTTPS on the API Management APIs.

Storing secrets

Never check passwords, access keys, or connection strings into source control. If these values are required, secure and deploy these values by using the appropriate techniques.

If a logic app works with sensitive data, see [Secure access and data for workflows in Azure Logic Apps](#) for detailed guidance.

API Management manages secrets by using objects called *named values* or *properties*. These objects securely store values that you can access through API Management policies. For more information, see [How to use Named Values in Azure API Management policies](#).

Cost Optimization

Cost Optimization is about looking at ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

In general, use the [Azure pricing calculator](#) to estimate costs. Here are some other considerations.

API Management

You're charged for all API Management instances when they're running. If you have scaled up and don't need that level of performance all the time, manually scale down or configure [autoscaling](#).

Logic Apps

Logic Apps uses a serverless model. Billing is calculated based on action and connector execution. For more information, see [Logic Apps pricing](#).

For more information, see the cost section in [Microsoft Azure Well-Architected Framework](#).

Operational Excellence

Operational Excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Design review checklist for Operational](#)

DevOps

Create separate resource groups for production, development, and test environments. Separate resource groups make it easier to manage deployments, delete test deployments, and assign access rights.

When you assign resources to resource groups, consider these factors:

- **Lifecycle.** In general, put resources that have the same lifecycle in the same resource group.
- **Access.** To apply access policies to the resources in a group, you can use [Azure RBAC](#).
- **Billing.** You can view rollup costs for the resource group.
- **Pricing tier for API Management.** Use the Developer tier for development and test environments. To minimize costs during preproduction, deploy a replica of your production environment, run your tests, and then shut down.

Deployment

Use [Azure Resource Manager templates](#) to deploy the Azure resources, follow the infrastructure as Code (IaC) Process. Templates make it easier to automate deployments using [Azure DevOps Services](#), or other CI/CD solutions.

Staging

Consider staging your workloads, which means deploying to various stages and running validations at each stage before moving on to the next one. If you use this approach, you can push updates to your production environments in a highly controlled way and minimize unanticipated deployment issues. [Blue-green deployment](#) and [Canary releases](#) are recommended deployment strategies for updating live production environments. Also consider having a good rollback strategy for when a deployment fails. For example, you could automatically redeploy an earlier, successful deployment from your deployment history. The `--rollback-on-error` flag parameter in Azure CLI is a good example.

Workload isolation

Put API Management and any individual logic apps in their own separate Resource Manager templates. By using separate templates, you can store the resources in source control systems.

You can deploy the templates together or individually as part of a CI/CD process.

Versions

Each time you change a logic app's configuration or deploy an update through a Resource Manager template, Azure keeps a copy of that version and keeps all versions that have a run history. You can use these versions to track historical changes or promote a version as the logic app's current configuration. For example, you can roll back a logic app to a previous version.

API Management supports two distinct but complementary versioning concepts:

- *Versions* allow API consumers to choose an API version based on their needs, for example, v1, v2, beta, or production.
- *Revisions* allow API administrators to make nonbreaking changes in an API and deploy those changes, along with a change log to inform API consumers about the changes.

You can make a revision in a development environment and deploy that change in other environments by using Resource Manager templates. For more information, see [Publish multiple versions of your API](#)

You can also use revisions to test an API before making the changes current and accessible to users. However, this method isn't recommended for load testing or integration testing. Use separate test or preproduction environments instead.

Diagnostics and monitoring

Use [Azure Monitor](#) for operational monitoring in both API Management and Logic Apps. Azure Monitor provides information based on the metrics configured for each service and is enabled by default. For more information, see:

- [Monitor published APIs](#)
- [Monitor status, set up diagnostics logging, and turn on alerts for Azure Logic Apps](#)

Each service also has these options:

- For deeper analysis and dashboarding, send Logic Apps logs to [Azure Log Analytics](#).
- For DevOps monitoring, configure Azure Application Insights for API Management.
- API Management supports the [Power BI solution template for custom API analytics](#). You can use this solution template for creating your own analytics solution. For business users, Power BI makes reports available.

Performance Efficiency

Performance Efficiency is the ability of your workload to scale to meet the demands placed on it by users in an efficient manner. For more information, see [Design review checklist for Performance Efficiency](#).

To increase the scalability of API Management, add [caching policies](#) where appropriate. Caching also helps reduce the load on back-end services.

To offer greater capacity, you can scale out Azure API Management Basic, Standard, and Premium tiers in an Azure region. To analyze the usage for your service, select **Capacity Metric** on the **Metrics** menu and then scale up or scale down as appropriate. The upgrade or scale process can take from 15 to 45 minutes to apply.

Recommendations for scaling an API Management service:

- Consider traffic patterns when scaling. Customers with more volatile traffic patterns need more capacity.
- Consistent capacity that's greater than 66% might indicate a need to scale up.
- Consistent capacity that's under 20% might indicate an opportunity to scale down.
- Before you enable the load in production, always load-test your API Management service with a representative load.

With the Premium tier, you can scale an API Management instance across multiple Azure regions. This makes API Management eligible for a higher SLA and lets you provision services near users in multiple regions.

The Logic Apps serverless model means administrators don't have to plan for service scalability. The service automatically scales to meet demand.

Next steps

- [Logic Apps](#)
- [API Management](#)
- [Azure DNS](#)

Related resources

For greater reliability and scalability, use message queues and events to decouple the backend systems. This architecture is shown in the next article in this series:

Enterprise integration using message queues and events

You might also be interested in these articles from the Azure Architecture Center:

- [Azure API Management landing zone architecture](#)
- [On-premises data gateway for Azure Logic Apps](#)

Use a message broker and events to integrate enterprise systems

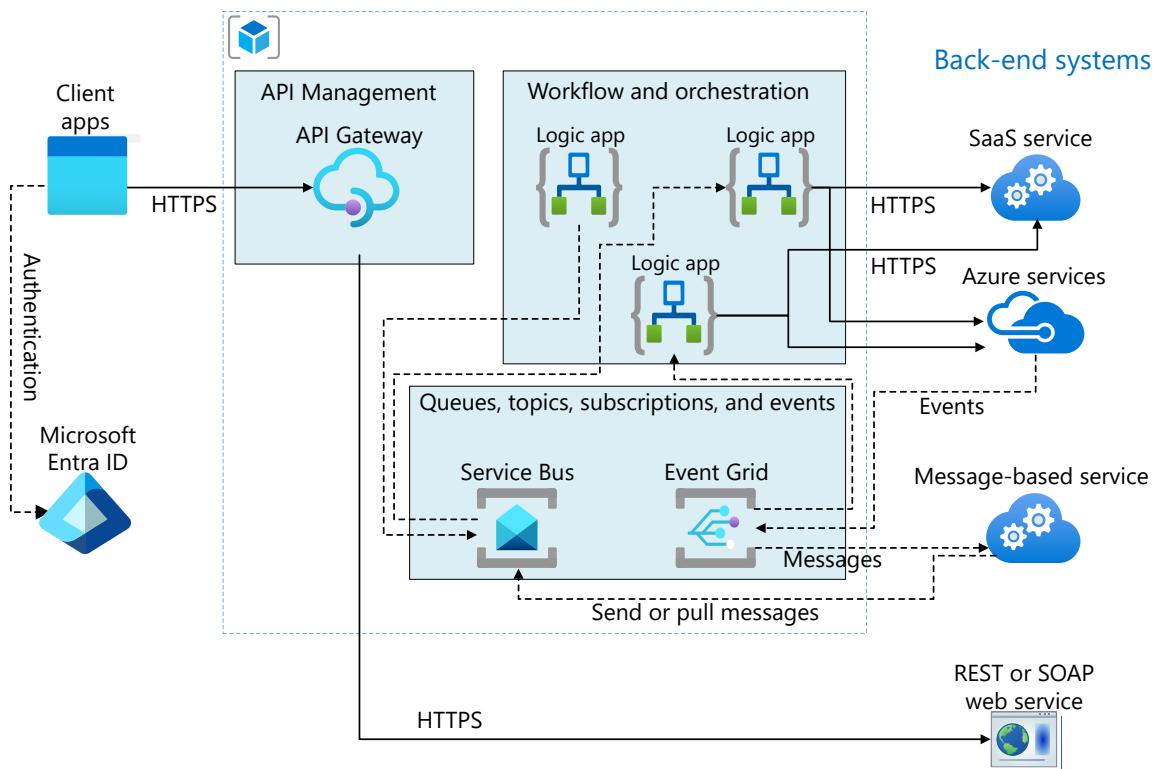
Azure Event Grid

Azure Service Bus

This architecture is based on the [basic enterprise integration](#) architecture but includes how to integrate enterprise back-end systems. This architecture uses message brokers and events to decouple services for greater scalability and reliability. Ensure that you're familiar with the design and components in the basic integration architecture. These elements provide foundational information about the core components of this architecture.

Architecture

The back-end systems that this design references include software as a service (SaaS) systems, Azure services, message-based services, and existing web services in your enterprise.



 Microsoft Azure



Download a [Visio file](#) of this architecture.

Scenario details

The preceding architecture builds on the simpler [basic enterprise integration architecture](#) that uses [Azure Logic Apps](#) to orchestrate workflows directly with back-end systems and uses [Azure API Management](#) to create catalogs of APIs.

This version of the architecture adds two components that help make the system more reliable and scalable:

- [Azure Service Bus](#) is a secure, reliable message broker.
- [Azure Event Grid](#) is an event-routing service. It uses a [publish and subscribe](#) eventing model.

This architecture uses asynchronous communication via a message broker instead of making direct, synchronous calls to back-end services. Asynchronous communication provides the following advantages:

- Uses the [Queue-Based Load Leveling pattern](#) to handle bursts in workloads via load-leveling
- Uses the [Publisher-Subscriber pattern](#) so that you can broadcast messages to multiple consumers
- Tracks the progress of long-running workflows reliably, even when they involve multiple steps or multiple applications
- Helps to decouple applications
- Integrates with existing message-based systems
- Provides the ability to queue messages when a back-end system isn't available

Use Event Grid so that various components in the system can react to events when they happen, rather than relying on polling or scheduled tasks. Similar to a message queue and topics, Event Grid helps decouple applications and services. If an application or service publishes events, any interested subscribers are notified. You can add new subscribers without updating the sender.

Many Azure services support sending events to Event Grid. For example, a logic app can listen for an event when new files are added to a blob store. This pattern creates reactive workflows in which uploading a file or putting a message on a queue starts a series of processes. The processes might run in parallel or in a specific sequence.

Recommendations

Consider the following recommendations. For more recommendations, see [Basic enterprise integration architecture](#).

Service Bus

Service Bus has two delivery models, the *pull* model and the *proxied push* model:

- **Pull model:** The receiver continuously polls for new messages. If you need to manage multiple queues and polling times, polling might be inefficient. But this model can simplify your architecture because it removes extra components and data hops.
- **Proxied push model:** The receiver initially subscribes to a specific event type on an Event Grid topic. When a new message is available, Service Bus raises and sends an event through Event Grid. This event then triggers the receiver to pull the next batch of messages from Service Bus. This model allows systems to receive messages almost in real time but without using resources to continuously poll for new messages. This architecture uses extra components that you must deploy, manage, and secure.

When you create a Standard Logic Apps workflow that consumes Service Bus messages, we recommend that you use the Service Bus built-in connector triggers. The built-in connector triggers abstract most of the pull model configuration without adding extra cost. This capability provides the right balance between cost, surface area management, and security because the connector continuously loops within the Logic Apps runtime engine. For more information, see [Service Bus built-in connector triggers](#).

Use [PeekLock mode](#) to access a group of messages. When you use PeekLock, the logic app can perform steps to validate each message before completing or abandoning the message. This approach prevents accidental message loss.

Event Grid

When an Event Grid trigger fires, it means that *at least one* event happened. For example, when a logic app gets an Event Grid trigger for a Service Bus message, there might be several messages available to process.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

Reliability

Reliability helps ensure that your application can meet the commitments that you make to your customers. For more information, see [Design review checklist for Reliability](#).

- **Microsoft Entra ID** is a globally distributed, highly available SaaS platform.
- You can deploy **API Management** in several highly available configurations, according to business requirements and cost tolerance. For more information, see [Ensure API Management availability and reliability](#).
- The **Logic Apps** Consumption tier supports geo-redundant storage. For more information, see [Business continuity and disaster recovery for Logic Apps](#).
- **Event Grid** resource definitions for topics, system topics, domains, and event subscriptions and event data are automatically replicated across [availability zones](#) in a region. When there's a failure in one of the availability zones, Event Grid resources automatically fail over to another availability zone without any human intervention. For more information, see [Cross-region disaster recovery and business continuity](#).
- **Service Bus** Premium supports [geo-disaster recovery](#) and [availability zones](#). Service Bus Standard supports [replication](#).

For information about guaranteed availability details of each service, see [SLAs for online services](#).

Security

Security provides assurances against deliberate attacks and the misuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

To help secure Service Bus, pair [Microsoft Entra authentication](#) with [managed identities](#). Microsoft Entra ID integration for Service Bus resources provides Azure role-based access control (RBAC) for fine-grained control over a client's access to resources. You can use Azure RBAC to grant permissions to a security principal, such as a user, a group, or an application service principal. The application service principal in this scenario is a managed identity.

If you can't use Microsoft Entra ID, use [shared access signature \(SAS\) authentication](#) to grant [users access and specific rights](#) to Service Bus resources.

If you need to expose a Service Bus queue or topic as an HTTP endpoint, for example, to post new messages, use API Management to help secure the queue by fronting the endpoint. You can then use certificates or OAuth authentication to help secure the endpoint. The easiest way

to help secure an endpoint is to use a logic app that has an HTTP request or response trigger as an intermediary.

The Event Grid service helps secure event delivery through a validation code. If you use Logic Apps to consume the event, validation is automatic. For more information, see [Event Grid security and authentication](#).

Network security

Consider network security throughout your design.

- You can bind [Service Bus Premium](#) to a virtual network subnet service endpoint. This configuration helps secure the namespace because it only accepts traffic from authorized virtual networks. You can also use [Azure Private Link](#) to only allow private traffic to your virtual network via [private endpoints](#).
- You can configure [Logic Apps Standard and Premium](#) to accept inbound traffic through [private endpoints](#) and to send outbound traffic through [virtual network integration](#).
- You can use an Azure virtual network to help secure access to your API Management instance and APIs. This method supports [private endpoints](#). For more information, see [Use a virtual network with API Management](#).

Cost Optimization

Cost Optimization focuses on ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

Use the [Azure pricing calculator](#) to estimate costs. Here are some other considerations.

API Management

You're charged for all API Management instances when they run. If you scale up, and then you no longer need that level of performance, manually scale down or configure [autoscaling](#).

For light usage workloads, consider the [Consumption tier](#), which is a low-cost, serverless option. The Consumption tier is billed per API call. Other tiers are billed per hour.

Logic Apps

Logic Apps uses a [serverless model](#). Billing is calculated based on the number of actions and connector calls. For more information, see [Logic Apps pricing](#).

Service Bus queues, topics, and subscriptions

Service Bus queues and subscriptions support both proxied push and pull models to deliver messages. In the pull model, every polling request is metered as an action. Even if you set long polling to the default of 30 seconds, cost can be high. Unless you need real-time message delivery, consider using the proxied push model.

Service Bus queues are included in all tiers: Basic, Standard, and Premium. Service Bus topics and subscriptions are available in Standard and Premium tiers. For more information, see [Service Bus pricing](#).

Event Grid

Event Grid uses a serverless model. Billing is calculated based on the number of operations. Operations include events that go to domains or topics, advanced matches, delivery attempts, and management calls. Usage of up to 100,000 operations is free of charge.

For more information, see [Event Grid pricing](#) and [Well-Architected Framework Cost Optimization](#).

Operational Excellence

Operational Excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Design review checklist for Operational Excellence](#).

The basic enterprise integration reference architecture provides [guidance about DevOps patterns](#), which align to the Well-Architected Framework [Operational Excellence](#) pillar.

Automate recovery operations as much as possible to help improve operational excellence. With automation in mind, you can combine [Azure log monitoring](#) with [Azure Automation](#) to automate the failover of your Service Bus resources. For an example of automation logic to initiate a failover, see [Failover flow](#).

Performance Efficiency

Performance Efficiency refers to your workload's ability to scale to meet user demands efficiently. For more information, see [Design review checklist for Performance Efficiency](#).

To achieve higher scalability, the Service Bus Premium tier can scale out the number of messaging units. For more information, see [Service Bus Premium and Standard messaging tiers](#) and [Autoscaling feature](#).

For more Service Bus recommendations, see [Best practices for performance improvements by using Service Bus messaging](#).

Next steps

- [Service Bus to Event Grid integration overview](#)
- [Tutorial that uses messaging to integrate non-Microsoft systems via NServiceBus ↗](#)

Related resources

- [Basic enterprise integration on Azure](#)
- [Enterprise business intelligence](#)

Build real-time monitoring and observable systems for media

Azure Data Explorer

Azure Functions

Azure AI Metrics Advisor

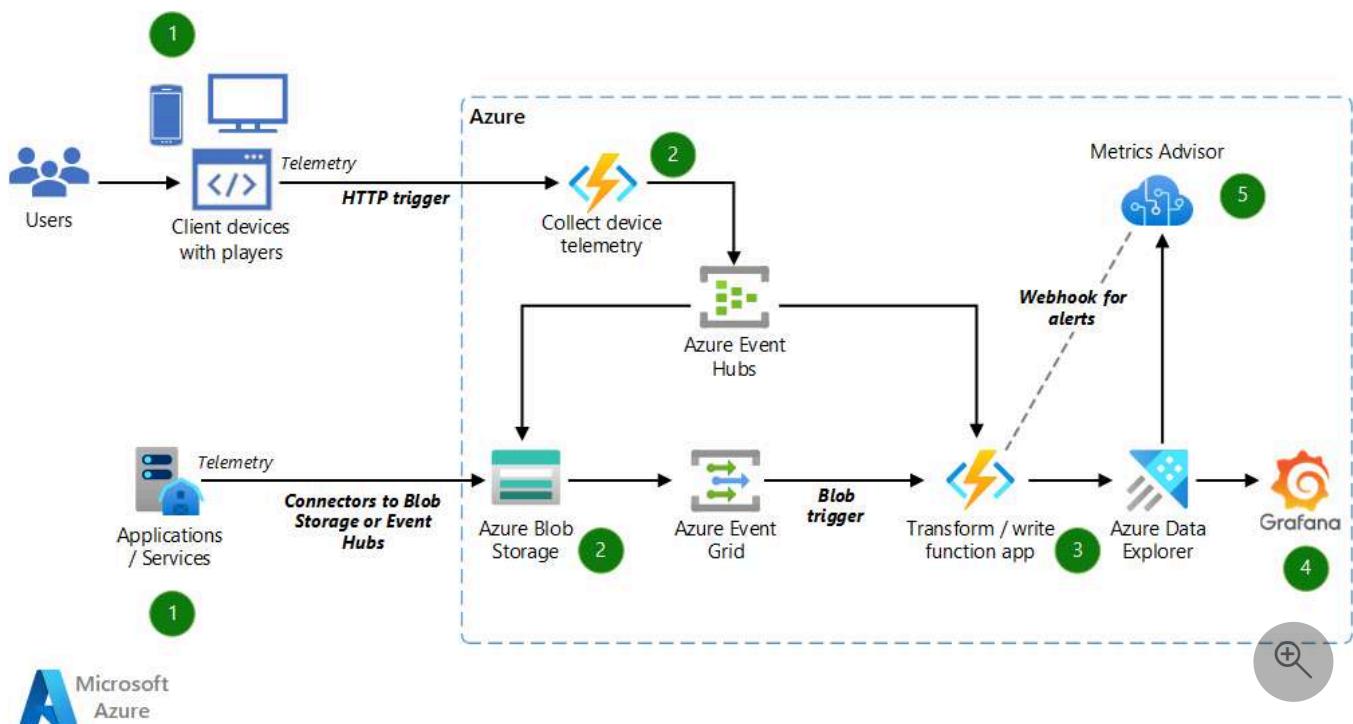
Azure Blob Storage

Azure Event Hubs

This architecture describes a solution that provides near real time monitoring and observability of systems and end-user device telemetry data. It focuses on a use case for the media industry.

Grafana [↗](#) is a trademark of its respective company. No endorsement is implied by the use of this mark.

Architecture



Download a [Visio file](#) [↗](#) of this architecture.

Dataflow

In the observable system shown in the diagram, raw telemetry is streamed to Azure Blob Storage via HTTP and connectors. The raw telemetry is processed, transformed, normalized, and saved in Azure Data Explorer for analysis. Systems like Grafana and Azure Metrics Advisor read data from Data Explorer and provide insights to end users.

More specifically, these are the elements of the system in the diagram:

- 1. Instrumentation.** Instrumentation occurs via probes or agents that are installed in systems to monitor data. These agents come in various forms. For example, in a video-on-

demand streaming platform, a company might use open-standards [dash.js](#) to collect Quality of Experience metrics from customers.

2. Ingestion. This raw telemetry can come directly from end clients via HTTP calls.

Alternatively, you can upload it via third-party systems to persistent storage and data lakes like Blob Storage. Blob Storage supports the ability to invoke an Azure function when a new file is uploaded. You can use this trigger mechanism to move raw telemetry to structured data warehouses.

3. Transformation and persistence. You might need a transformation system to normalize your data. An Azure Functions app transforms the data as needed and then writes it to Data Explorer. Data Explorer is ideal for big data analytics because it's designed for high performance and throughput on large data sets.

4. Monitoring. Azure Managed Grafana supports integration with Data Explorer. You can use the drag-and-drop features of Grafana to quickly build dashboards and charts.

Grafana is a good fit for media monitoring because it provides sub-minute refreshing of dashboard tiles and can also be used for alerting.

5. Anomaly detection. The Grafana dashboard provides support for manual monitoring in the NOC. However, with a large data set and a user base spread across geographies and using various devices, manual identification of issues via charts and alert rules that have hard-coded thresholds becomes inefficient. You can use AI to address this problem. Services like Metrics Advisor use machine learning algorithms to automatically understand and detect anomalies based on time-series data. In addition, the Kusto data platform has built-in anomaly detection functions that account for seasonality and baseline trends in the data.

Components

- [Azure Data Explorer](#) is a managed data analytics service for near real-time analysis of large volumes of data. Azure Data Explorer is a tool for handling large datasets that require high speed and throughput of data retrieval. In this architecture, Azure Data Explorer is used to store and query datasets for analysis.
- [Blob Storage](#) is a cloud storage service for unstructured data. This telemetry can come from your applications and services or from non-Microsoft vendors. In this architecture, Blob Storage is the initial landing zone for raw telemetry data from applications, services, or partner vendors. You can treat this data as transient if you don't need to perform more analysis later. The data from Blob Storage is ingested into Azure Data Explorer clusters.
- [Azure Event Grid](#) is an event delivery system that routes events from publishers to subscribers. In this architecture, Event Grid listens to events that Blob Storage publishes. Azure Storage events allow applications to react to events such as the creation and deletion of blobs. An Azure function subscribes to events that Event Grid publishes.

- [Azure Event Hubs](#) is a streaming data ingestion service that can ingest millions of events per second from any source. In this architecture, it serves as the front door, or *event ingestor*, for an event pipeline. An event ingestor is a component or service that's located between event publishers and event consumers. It decouples the production of an event stream from the consumption of the events.
- [Azure Functions](#) is a serverless solution that's used to parse and transform data. In this architecture, Azure Functions processes raw telemetry ingested via HTTP and blob endpoints and writes it to the Azure Data Explorer cluster for analysis.
- [Azure Managed Grafana](#) is a managed service that provides Grafana dashboards and visualization capabilities. In this architecture, it connects to Azure Data Explorer to generate charts and dashboards that visualize telemetry data. Azure Managed Grafana provides deep integration with Microsoft Entra ID so that you can implement role-based access to dashboards and views.
- [Metrics Advisor](#) is a part of Azure AI services. It uses AI to perform data monitoring and anomaly detection in time-series data. In this architecture, Metrics Advisor automates the process of applying models to data. It also provides a set of APIs and a web-based workspace for data ingestion, anomaly detection, and diagnostics. You can use it even if you have no knowledge of machine learning.

Alternatives

[Azure Data Factory](#) and [Azure Synapse Analytics](#) provide tools and workspaces for building ETL workflows and the ability to track and retry jobs from a graphical interface. Note that Data Factory and Azure Synapse both have a minimum lag of about 5 minutes from the time of ingestion to persistence. This lag might be acceptable in your monitoring system. If it is, we recommend that you consider these alternatives.

Scenario details

Organizations often deploy varied and large-scale technologies to solve business problems. These systems, and end-user devices, generate large sets of telemetry data.

This architecture is based on a use case for the media industry. Media streaming for live and video-on-demand playback requires near real-time identification of and response to application problems. To support this real-time scenario, organizations need to collect a massive telemetry set, which requires scalable architecture. After the data is collected, other types of analysis, like AI and anomaly detection, are needed to efficiently identify problems across so large a data set.

When large-scale technologies are deployed, the system and end-user devices that interact with them generate massive sets of telemetry data. In traditional scenarios, this data is analyzed via a data warehouse system to generate insights that can be used to support management decisions. This approach might work in some scenarios, but it's not responsive enough for streaming media use cases. To solve this problem, real-time insights are required for the telemetry data that's generated from monitoring servers, networks, and the end-user devices that interact with them. Monitoring systems that catch failures and errors are common, but to catch them in near real-time is difficult. That's the focus of this architecture.

In a live streaming or video-on-demand setting, telemetry data is generated from systems and heterogeneous clients (mobile, desktop, and TV). The solution involves taking raw data and associating context with the data points, for example, dimensions like geography, end-user operating system, content ID, and CDN provider. The raw telemetry is collected, transformed, and saved in Data Explorer for analysis. You can then use AI to make sense of the data and automate the manual processes of observation and alerting. You can use systems like Grafana and Metrics Advisor to read data from Data Explorer to show interactive dashboards and trigger alerts.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Microsoft Azure Well-Architected Framework](#).

Reliability

Reliability ensures your application can meet the commitments you make to your customers. For more information, see [Design review checklist for Reliability](#).

Business-critical applications need to keep running even during disruptive events like Azure region or CDN outages. There are two primary strategies and one hybrid strategy for building redundancy into your system:

- **Active/active.** Duplicate code and functions are running. Either system can take over during a failure.
- **Active/standby.** Only one node is active/primary. The other one is ready to take over in case the primary node goes down.
- **Mixed.** Some components/services are in the active/active configuration, and some are in active/standby.

Keep in mind that not all Azure services have built-in redundancy. For example, Azure Functions runs a function app only in a specific region. [Azure Functions geo-disaster recovery](#)

describes various strategies that you can implement, depending on how your functions are triggered (HTTP versus pub/sub).

The ingestion and transformation function app can run in active/active mode. You can run Data Explorer in both [active/active and active/standby configurations](#).

Azure Managed Grafana supports [availability zone redundancy](#). One strategy for creating cross-region redundancy is to set up Grafana in each region in which your Data Explorer cluster is deployed.

Cost Optimization

Cost Optimization is about looking at ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

The cost of this architecture depends on the number of ingress telemetry events, your storage of raw telemetry in Blob Storage and Data Explorer, an hourly cost for Azure Managed Grafana, and a static cost for the number of time-series charts in Metrics Advisor.

You can use the [Azure pricing calculator](#) to estimate your hourly or monthly costs.

Performance Efficiency

Performance Efficiency is the ability of your workload to scale to meet the demands placed on it by users in an efficient manner. For more information, see [Design review checklist for Performance Efficiency](#).

Depending on the scale and frequency of incoming requests, the function app might be a bottleneck, for two main reasons:

- **Cold start.** Cold start is a consequence of serverless executions. It refers to the scheduling and setup time that's required to spin up an environment before the function first starts running. At most, the required time is a few seconds.
- **Frequency of requests.** Say you have 1,000 HTTP requests but only a single-threaded server to handle them. You won't be able to service all 1,000 HTTP requests concurrently. To serve these requests in a timely manner, you need to deploy more servers. That is, you need to scale horizontally.

We recommend that you use Premium or Dedicated SKUs to:

- Eliminate cold start.

- Handle requirements for concurrent requests by scaling the number of servicing virtual machines up or down.

For more information, see [Select a SKU for your Azure Data Explorer cluster](#).

Deploy this scenario

For information about deploying this scenario, see [real-time-monitoring-and-observability-for-media](#) on GitHub. This code sample includes the necessary infrastructure-as-code (IaC) to bootstrap development and Azure functions to ingest and transform the data from HTTP and blob endpoints.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal authors:

- [John Hauppa](#) | Senior Technical Program Manager
- [Uffaz Nathaniel](#) | Principal Software Engineer

Other contributors:

- [Dilmurod Makhmadaliev](#) | Software Engineer
- [Omeed Musavi](#) | Principal Software Engineer Lead
- [Ayo Mustapha](#) | Principal Technical Program Manager

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

- [Supplementary code samples](#)
- [Azure Data Explorer documentation](#)
- [Introduction to Azure Data Explorer - Training](#)
- [Introduction to Azure Functions](#)

Related resources

- [Monitor Media Services](#)
- [Analytics architecture design](#)

Enable machine learning inference on an Azure IoT Edge device

Azure IoT Edge

Azure IoT Hub

AI on the edge is one of the most popular edge scenarios. Implementations of this scenario include image classification, object detection, body, face, and gesture analysis, and image manipulation. This architecture guide describes how to use Azure IoT Edge to support these scenarios.

You can improve AI accuracy by updating the AI model, but in some scenarios the edge device network environment isn't good. For example, in the wind power and oil industries, equipment might be located in the desert or the ocean.

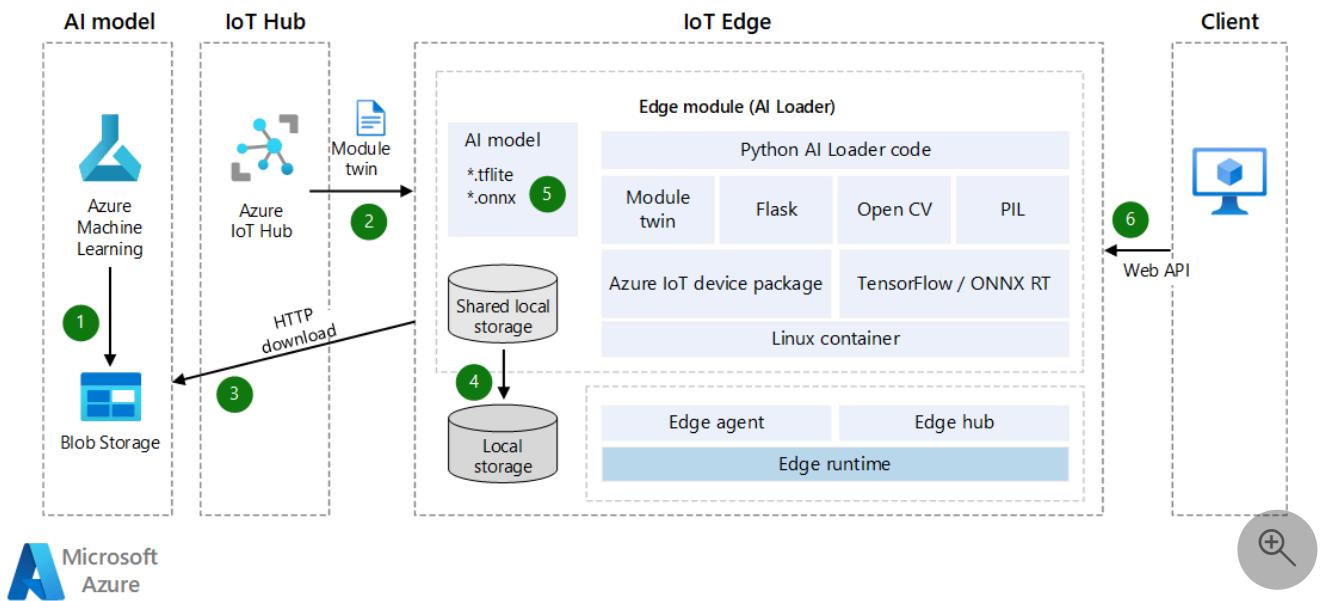
IoT Edge module twins are used to implement the dynamically loaded AI model. IoT Edge modules are based on Docker. An image for an IoT Edge module in an AI environment typically has a size of at least 1 GB, so incrementally updating the AI model is important in a narrow-bandwidth network. That consideration is the main focus of this article. The idea is to create an IoT Edge AI module that can load LiteRT (formerly TensorFlow Lite) or Open Neural Network Exchange (ONNX) object detection models. You can also enable the module as a web API so that you can use it to benefit other applications or modules.

The solution described in this article can help you in these ways:

- Enable AI inference on edge devices.
- Minimize the network cost of deploying and updating AI models on the edge. The solution can save money for you or your customers, especially in a narrow-bandwidth network environment.
- Create and manage an AI model repository in an IoT edge device's local storage.
- Achieve almost zero downtime when the edge device switches AI models.

TensorFlow and LiteRT  are a trademark of Google Inc. No endorsement is implied by the use of this mark.

Architecture



Download a [Visio file](#) of this architecture.

Dataflow

1. The AI model is uploaded to Azure Blob Storage or a web service. The model can be a pre-trained LiteRT or ONNX model or a model created in Azure Machine Learning. The IoT Edge module can access this model and download it to the edge device later. If you need better security, consider using private endpoint connections between Blob Storage and the edge device.
2. Azure IoT Hub syncs device module twins automatically with AI model information. The sync occurs even if IoT Edge has been offline. (In some cases, IoT devices are connected to networks at scheduled hourly, daily, or weekly times to save power or reduce network traffic.)
3. The loader module monitors the updates of the module twins via API. When it detects an update, it gets the machine learning model SAS token and then downloads the AI model.
 - For more information, see [Create SAS token for a container or blob](#).
 - You can use the **ExpiresOn** property to set the expiration date of resources. If your device will be offline for a long time, you can extend the expiration time.
4. The loader module saves the AI model in the shared local storage of the IoT Edge module. You need to configure the shared local storage in the IoT Edge deployment JSON file.
5. The loader module loads the AI model from local storage via the LiteRT or ONNX API.
6. The loader module starts a web API that receives the binary photo via POST request and returns the results in a JSON file.

To update the AI model, you can upload the new version to Blob Storage and sync the device module twins again for an incremental update. There's no need to update the whole IoT Edge module image.

Scenario details

In this solution, an IoT Edge module is used to download an AI model and then enable machine learning inference. You can use pre-trained LiteRT or ONNX models in this solution.

LiteRT

- A `.tflite` file is a pre-trained AI model. You can download one from [TensorFlow.org](#). It's a generic AI model that you can use in cross-platform applications like iOS and Android. LiteRT supports models from TensorFlow, PyTorch, JAX, and Keras. For more information about metadata and associated fields (for example, `labels.txt`) see [Read the metadata from models](#).
- An object detection model is trained to detect the presence and location of multiple classes of objects. For example, a model might be trained with images that contain various pieces of fruit, along with a label that specifies the class of fruit that they represent (for example, apple) and data that specifies where each object appears in the image.

When an image is provided to the model, it outputs a list of the objects that it detects, the location of a bounding box for each object, and a score that indicates the confidence of the detection.

- If you want to build or custom-tune an AI model, see [LiteRT Model Maker](#).
- You can get more free pre-trained detection models, with various latency and precision characteristics, at [Detection Zoo](#). Each model uses the input and output signatures shown in the following code samples.

ONNX

ONNX is an open-standard format for representing machine learning models. It's supported by a community of partners who have implemented it in many frameworks and tools.

- ONNX supports tools for building and deploying models and for accomplishing other tasks. For more information, see, [Supported ONNX tools](#).
- You can use ONNX Runtime to run ONNX pre-trained models. For information about pre-trained models, see [ONNX Model Zoo](#).

- For this scenario, you can use an object detection and image segmentation model: [Tiny YOLOv3](#).

The ONNX community [provides tools](#) to help you create and deploy your deep learning model.

Download trained AI models

To download trained AI models, we recommend that you use device twins to receive notifications when a new model is ready. Even if the device is offline, the message can be cached in IoT Hub until the edge device comes back online. The message will be synchronized automatically.

Following is an example of Python code that registers notifications for the device twins and then downloads the AI model in a ZIP file. It also performs further operations on the downloaded file.

The code performs these tasks:

1. Receive the device twins notification. The notification includes the file name, file download address, and MD5 authentication token. (In the file name, you can include version information, like 1.0.)
2. Download the AI model as a ZIP file to local storage.
3. Optionally, perform MD5 checksum. MD5 verification helps prevent ZIP files that have been tampered with during network transmission.
4. Unzip the ZIP file and save it locally.
5. Send a notification to IoT Hub or a routing message to report that the new AI model is ready.

Python

```
# define behavior for receiving a twin patch
async def twin_patch_handler(patch):
    try:
        print( "##### The data in the desired properties patch was: %s" %
patch)
        if "FileName" in patch:
            FileName = patch["FileName"]
        if "DownloadUrl" in patch:
            DownloadUrl = patch["DownloadUrl"]
        if "ContentMD5" in patch:
            ContentMD5 = patch["ContentMD5"]
            FilePath = "/iotedge/storage/" + FileName

        # download AI model
        r = requests.get(DownloadUrl)
```

```

print ("##### download AI Model Succeeded.")
ffw = open(FilePath, 'wb')
ffw.write(r.content)
ffw.close()
print ("##### AI Model File: " + FilePath)

# MD5 checksum
md5str = content_encoding(FilePath)
if md5str == ContentMD5:
    print ("##### New AI Model MD5 checksum succeeded")
    # decompressing the ZIP file
    unZipSrc = FilePath
    targeDir = "/iotedge/storage/"
    filenamenoext = get_filename_and_ext(unZipSrc)[0]
    targeDir = targeDir + filenamenoext
    unzip_file(unZipSrc,targeDir)

    # ONNX
    local_model_path = targeDir + "/tiny-yolov3-11.onnx"
    local_labelmap_path = targeDir + "/coco_classes.txt"

    # LiteRT
    # local_model_path = targeDir +
"/ssd_mobilenet_v1_1_metadata_1.tflite"
    # local_labelmap_path = targeDir + "/labelmap.txt"

    # message to module
    if client is not None:
        print ("##### Send AI Model Info AS Routing Message")
        data = "{\"local_model_path\": \"%s\", \"local_labelmap_path\": \"%s\"}" % (filenamenoext+"/tiny-yolov3-11.onnx",
filenamenoext+"/coco_classes.txt")
        await client.send_message_to_output(data, "DLModelOutput")
        # update the reported properties
        reported_properties = {"LatestAIModelFileName": FileName }
        print("##### Setting reported LatestAIModelName to
{}".format(reported_properties["LatestAIModelFileName"]))
        await client.patch_twin_reported_properties(reported_properties)
    else:
        print ("##### New AI Model MD5 checksum failed")

except Exception as ex:
    print ( "Unexpected error in twin_patch_handler: %s" % ex )

```

Inference

After the AI model is downloaded, the next step is to use the model on the edge device. You can dynamically load the model and perform object detection on edge devices. The following code example shows how to use the LiteRT AI model to detect objects on edge devices.

The code performs these tasks:

1. Dynamically load the LiteRT AI model.
2. Perform image standardization.
3. Detect objects.
4. Compute detection scores.

Python

```
class InferenceProcedure():

    def detect_object(self, imgBytes):

        results = []
        try:
            model_full_path = AI_Model_Path.Get_Model_Path()
            if(model_full_path == ""):
                raise Exception ("PLEASE SET AI MODEL FIRST")
            if '.tflite' in model_full_path:
                interpreter = tf.lite.Interpreter(model_path=model_full_path)
                interpreter.allocate_tensors()
                input_details = interpreter.get_input_details()
                output_details = interpreter.get_output_details()
                input_shape = input_details[0]['shape']

                # bytes to numpy.ndarray
                im_arr = np.frombuffer(imgBytes, dtype=np.uint8)
                img = cv2.imdecode(im_arr, flags=cv2.IMREAD_COLOR)
                im_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
                im_rgb = cv2.resize(im_rgb, (input_shape[1], input_shape[2]))
                input_data = np.expand_dims(im_rgb, axis=0)

                interpreter.set_tensor(input_details[0]['index'], input_data)
                interpreter.invoke()
                output_data = interpreter.get_tensor(output_details[0]['index'])
                detection_boxes = interpreter.get_tensor(output_details[0]
                ['index'])
                detection_classes = interpreter.get_tensor(output_details[1]
                ['index'])
                detection_scores = interpreter.get_tensor(output_details[2]
                ['index'])
                num_boxes = interpreter.get_tensor(output_details[3]['index'])

                label_names = [line.rstrip('\n') for line in
open(AI_Model_Path.Get_Labelmap_Path())]
                label_names = np.array(label_names)
                new_label_names = list(filter(lambda x : x != '????', label_names))

                for i in range(int(num_boxes[0])):
                    if detection_scores[0, i] > .5:
                        class_id = int(detection_classes[0, i])
                        class_name = new_label_names[class_id]
                        # top, left, bottom, right
                        results_json = "{\"Class\": '%s', 'Score': '%s', 'Location': '%s'}" % (class_name, detection_scores[0, i],detection_boxes[0, i])
                        results.append(results_json)

        except Exception as e:
            print(e)
```

```

        results.append(results_json)
        print(results_json)
    except Exception as e:
        print ( "detect_object unexpected error %s " % e )
        raise

    # return results
    return json.dumps(results)

```

Following is the ONNX version of the preceding code. The steps are mostly the same. The only difference is how the detection score is handled, because the `Labelmap` and model output parameters are different.

Python

```

class InferenceProcedure():

    def letterbox_image(self, image, size):
        '''resize image with unchanged aspect ratio using padding'''
        iw, ih = image.size
        w, h = size
        scale = min(w/iw, h/ih)
        nw = int(iw*scale)
        nh = int(ih*scale)

        image = image.resize((nw,nh), Image.BICUBIC)
        new_image = Image.new('RGB', size, (128,128,128))
        new_image.paste(image, ((w-nw)//2, (h-nh)//2))
        return new_image

    def preprocess(self, img):
        model_image_size = (416, 416)
        boxed_image = self.letterbox_image(img, tuple(reversed(model_image_size)))
        image_data = np.array(boxed_image, dtype='float32')
        image_data /= 255.
        image_data = np.transpose(image_data, [2, 0, 1])
        image_data = np.expand_dims(image_data, 0)
        return image_data

    def detect_object(self, imgBytes):
        results = []
        try:
            model_full_path = AI_Model_Path.Get_Model_Path()
            if(model_full_path == ""):
                raise Exception ("PLEASE SET AI MODEL FIRST")
            if '.onnx' in model_full_path:

                # input
                image_data = self.preprocess(imgBytes)
                image_size = np.array([imgBytes.size[1], imgBytes.size[0]],

dtype=np.float32).reshape(1, 2)

```

```

labels_file = open(AI_Model_Path.Get_Labelmap_Path())
labels = labels_file.read().split("\n")

    # Loading ONNX model
    print("loading Tiny YOLO...")
    start_time = time.time()
    sess = rt.InferenceSession(model_full_path)
    print("loaded after", time.time() - start_time, "s")

    input_name00 = sess.get_inputs()[0].name
    input_name01 = sess.get_inputs()[1].name
    pred = sess.run(None, {input_name00:
image_data, input_name01:image_size})

    boxes = pred[0]
    scores = pred[1]
    indices = pred[2]

    results = []
    out_boxes, out_scores, out_classes = [], [], []
    for idx_ in indices[0]:
        out_classes.append(idx_[1])
        out_scores.append(scores[tuple(idx_)])
        idx_1 = (idx_[0], idx_[2])
        out_boxes.append(boxes[idx_1])
        results_json = "{\"Class\": '%s', 'Score': '%s', 'Location': '%s'}" % (labels[idx_[1]], scores[tuple(idx_)], boxes[idx_1])
        results.append(results_json)
        print(results_json)

    except Exception as e:
        print ( "detect_object unexpected error %s " % e )
        raise

    # return results
    return json.dumps(results)

```

If your IoT edge device incorporates the preceding code and features, your edge device has AI image object detection and supports dynamic update of AI models. If you want the edge module to provide AI functionality to other applications or modules via a web API, you can create a web API in your module.

Flask framework is one example of a tool that you can use to quickly create an API. You can receive images as binary data, use an AI model for detection, and then return the results in a JSON format. For more information, see [Flask: Flask Tutorial in Visual Studio Code](#).

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Bo Wang ↗](#) | Senior Software Engineer

Other contributor:

- [Freddy Ayala ↗](#) | Cloud Solution Architect

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

- [Understand and use module twins in IoT Hub](#)
- [Learn how to deploy modules and establish routes in IoT Edge](#)
- [Give modules access to a device's local storage](#)
- [Understand IoT Edge automatic deployments for single devices or at scale](#)
- [Open Neural Network Exchange ↗](#)
- [ONNX Tutorials ↗](#)
- [Deploy a machine learning model on IoT and edge devices ↗](#)

Related resources

- [IoT architectures](#)
- [Choose an IoT solution in Azure](#)
- [AI architecture design](#)

Scale out an Azure IoT Hub solution to support millions of devices

10/12/2025

This article describes how to scale an Internet of Things (IoT) solution by using a scale-out pattern on the Azure IoT Hub platform. The scale-out pattern addresses scaling challenges by adding instances to a deployment instead of increasing instance size. This implementation guidance shows you how to scale an IoT solution that supports millions of devices and considers Azure service and subscription limits. The article outlines low-touch and zero-touch deployment models of the scale-out pattern that you can adopt depending on your needs.



For more information, see the following articles:

- [Best practices for large-scale Microsoft Azure IoT device deployments](#)
- [IoT Hub](#)
- [IoT Hub Device Provisioning Service \(DPS\)](#)

(!) Note

This document doesn't cover the [Azure IoT Operations](#) platform, which scales based on the hosting Kubernetes platform configuration.

Gather requirements

Gather requirements before you implement a new IoT solution. This step helps ensure that the implementation meets your business objectives. Your business objectives and operational environment should drive your requirements. At a minimum, gather the following requirements:

Identify the types of devices that you want to deploy. IoT encompasses a wide range of solutions, from simple microcontroller units (MCUs) to mid-level system-on-chip (SoC) and microprocessor units (MPUs), to full PC-level designs. Device-side software capabilities directly influence the solution design.

Determine the number of devices that you need to deploy. Some basic principles of implementing IoT solutions apply at all scales. Understand the scale to help avoid overengineering a solution. A solution for 1,000 devices has fundamental differences compared

to a solution for 1 million devices. A proof-of-concept (PoC) solution for 10,000 devices might not scale appropriately to 10 million devices if you don't consider the target scale at the start of the solution design.

Identify how many devices you need to deploy so that you can choose the right Azure IoT service. The scaling for IoT Hub and IoT Hub DPS differs. By design, a single DPS instance can route to multiple IoT Hub instances. So consider the scale of each service individually with respect to the number of devices. But limits don't exist in isolation. If one service presents a limit concern, other services likely do too. Treat service limits as distinct but related quotas.

Document the anticipated device locations. Include physical location, power availability, and internet connectivity. A solution that you deploy in a single geography, such as only in North America, is designed differently compared to a global solution. Likewise, an industrial IoT solution deployed in factories that have full-time power differs from a fleet management solution deployed in motor vehicles that have variable power and location. The communication protocol and available bandwidth, whether to a gateway or directly to a cloud service, affect design scalability at each layer. Also consider connectivity availability. Determine whether devices remain connected to Azure or run in a disconnected mode for extended periods.

Investigate data locality requirements. Legal, compliance, or customer requirements might restrict where you can store data (such as telemetry) or metadata (such as device information) for the solution. These restrictions significantly influence the solution's geographical design.

Determine data exchange requirements. A solution that sends basic telemetry such as current temperature once per hour differs from a solution that uploads 1-MB sample files once every 10 minutes. A one-way, device-to-cloud (D2C) solution differs from a bidirectional D2C and cloud-to-device (C2D) solution. Also, product scalability limitations treat message size and message quantity as different dimensions.

Document expected high availability and disaster recovery requirements. Like any production solution, full IoT solution designs include availability, or uptime, requirements. The design needs to cover both planned maintenance scenarios and unplanned downtime, including user errors, environmental factors, and solution bugs. The design also needs to have a documented [recovery point objective \(RPO\)](#) and recovery time objective (RTO) if a disaster occurs, such as a permanent region loss or malicious actors. This article focuses on device scale, so it includes only limited information about high availability and disaster recovery concerns.

Decide on a customer tenancy model if appropriate. In a multitenant software development company solution, where the solution developer creates a solution for external customers, the design must define how to segregate and manage customer data. For more information, see [Tenancy models](#) and the related [IoT-specific guidance](#).

Understand Azure IoT concepts

When you create a solution, choose the appropriate Azure IoT components and other supporting Azure services. The architecture of your solution requires significant effort. Properly using the IoT Hub and IoT Hub DPS services can help you scale your solutions to millions of devices.

IoT Hub

IoT Hub is a managed service hosted in the cloud that acts as a central message hub for communication between an IoT application and its attached devices. You can use IoT Hub alone or with IoT Hub DPS.

IoT Hub scales based on the desired functionality and the number of messages or volume of data per day. Use the following three inputs to determine how to scale an instance:

- The free, basic, and standard [tiers](#) determine the available capabilities. A production instance doesn't use the free tier because it's limited in scale and intended for introduction development scenarios only. Most solutions use the standard tier to get the full capabilities of IoT Hub.
- The [size](#) determines the message and data throughput base unit for D2C messages for IoT Hub. The maximum size for an instance of IoT Hub is size 3, which supports 300 million messages per day and 1,114.4 GB of data per day, per unit.
- The unit count determines the multiplier for the scale on size. For example, three units support three times the scale of one unit. The limit on size 1 or 2 hub instances is 200 units, and the limit on size 3 hub instances is 10 units.

In addition to the daily limits based on the size and unit count and the general functionality limits based on tier, IoT Hub enforces per-second limits on throughput. Each IoT Hub instance also supports up to 1 million devices as a [hard limit](#). Your data exchange requirements help define the appropriate configuration. For more information, see [Other limits](#).

Your solution requirements drive the necessary size and number of IoT Hub instances as a starting point. If you use IoT Hub DPS, Azure helps you distribute your workloads across multiple IoT Hub instances.

IoT Hub DPS

IoT Hub DPS is a helper service for IoT Hub that enables zero-touch, just-in-time provisioning to the right IoT hub without requiring human intervention. Each Azure subscription supports a maximum of [10 DPS instances](#). Each service instance supports a maximum of [1 million](#)

[registrations](#). Address service limits in your workload design limit to avoid problems in the future.

DPS instances reside in specific geographic regions but have a [global public endpoint](#) by default. Specific instances are accessed through an ID scope. Because instances are in specific regions and each instance has its own ID scope, you should be able to configure the ID scope for your devices.

Understand shared resiliency concepts

You must consider shared resiliency concepts, such as transient fault handling, device location impact, and, for software companies, software as a service (SaaS) data resiliency.

Understand transient fault handling. Any production distributed solution, whether it's on-premises or in the cloud, must be able to recover from transient or temporary faults. Transient faults might occur more frequently in a cloud solution because of the following factors:

- Reliance on an external provider
- Reliance on the network connectivity between the device and cloud services
- Implementation limits of cloud services

Transient fault handling requires you to have a retry capability built into your device code. Several retry strategies exist, including exponential backoff with randomization, also known as *exponential backoff with jitter*. For more information, see [Transient fault handling](#).

Different factors can affect the network connectivity of a device:

- **The power source of a device:** Battery-powered devices or devices powered by transient sources, such as solar or wind, might have less network connectivity than full-time line-powered devices.
- **The deployment location of a device:** Devices in urban factory settings likely have better network connectivity than devices in isolated field environments.
- **The location stability of a device:** Mobile devices likely have less network connectivity than fixed-location devices.

These concerns also affect the timing of device availability and connectivity. For example, line-powered devices in dense, urban environments, such as smart speakers, might disconnect and reconnect in large groups. Consider the following scenarios:

- A blackout might cause 1 million devices to go offline at the same time and come back online simultaneously because of power grid loss and reconnection. This scenario applies in both consumer scenarios, such as smart speakers, and business and industrial IoT

scenarios, such as connected, line-powered thermostats reporting to a real-estate management company.

- During a short-timeframe, large-scale onboarding event, such as [Black Friday](#) or Christmas, many consumers power on devices for the first time in a relatively short period of time.
- Many devices receive scheduled updates in a short time window, and all of them reboot with the new update at approximately the same time.

These *many devices booting at once* scenarios can trigger cloud service throttling, even with near-constant network connectivity.

Beyond network and quota problems, you should also consider Azure service outages. These outages might affect individual services or entire regions. Some services, such as IoT Hub, support geo-redundancy. Other services, such as IoT Hub DPS, store their data in a single region. You can link one IoT hub to multiple DPS instances, which helps mitigate regional risks.

If regional redundancy is a concern, use the [Geode pattern](#). This pattern hosts independent, grouped resources across different geographies. Similarly, a deployment stamp, also known as a *scale stamp*, applies this pattern to operate multiple workloads or tenants. For more information, see [Deployment Stamps pattern](#).

Understand device location impact. Most Azure services are [regional](#), even DPS with global endpoints. Exceptions include Azure Traffic Manager and Microsoft Entra ID. Your decisions about device location, data location, and metadata location (such as Azure resource groups) play a critical role in your design.

- **Device location:** Device location requirements affect your regional selection because they affect transactional latency.
- **Data location:** Data location is tied to device location, which is subject to compliance concerns. For example, a solution that stores data for a state in the United States might require data storage in the US [geography](#). Data locality requirements might also drive this need.
- **Metadata location:** Although device location doesn't usually affect metadata location because devices interact with solution *data* and not solution *metadata*, compliance and cost concerns affect metadata location. In many cases, convenience dictates that the metadata location is the same as the data location for regional services.

The Azure Cloud Adoption Framework includes [guidance about regional selection](#).

Understand software company SaaS concerns. Software companies that offer SaaS solutions should meet customers' expectations for availability and resiliency. Software companies must

architect Azure services to be highly available and consider the cost of resiliency and redundancy when billing the customer.

Segregate the cost of goods sold based on customer data segregation for each software customer. This distinction is important when the user isn't the same as the customer. For example, for a smart TV platform, the platform vendor's customer might be the television vendor, but the user is the purchaser of the television. This segregation, driven by the customer tenancy model from the requirements, requires separate DPS and IoT Hub instances. The provisioning service must also have a unique customer identity, which you can define through a unique endpoint or device authentication process. For more information, see [IoT multitenant guidance](#).

Scale out components and their supporting services

When you scale IoT solutions, evaluate each service and how they interrelate. You can scale your IoT solution across multiple DPS instances or by using IoT Hub.

Scale out across multiple DPS instances

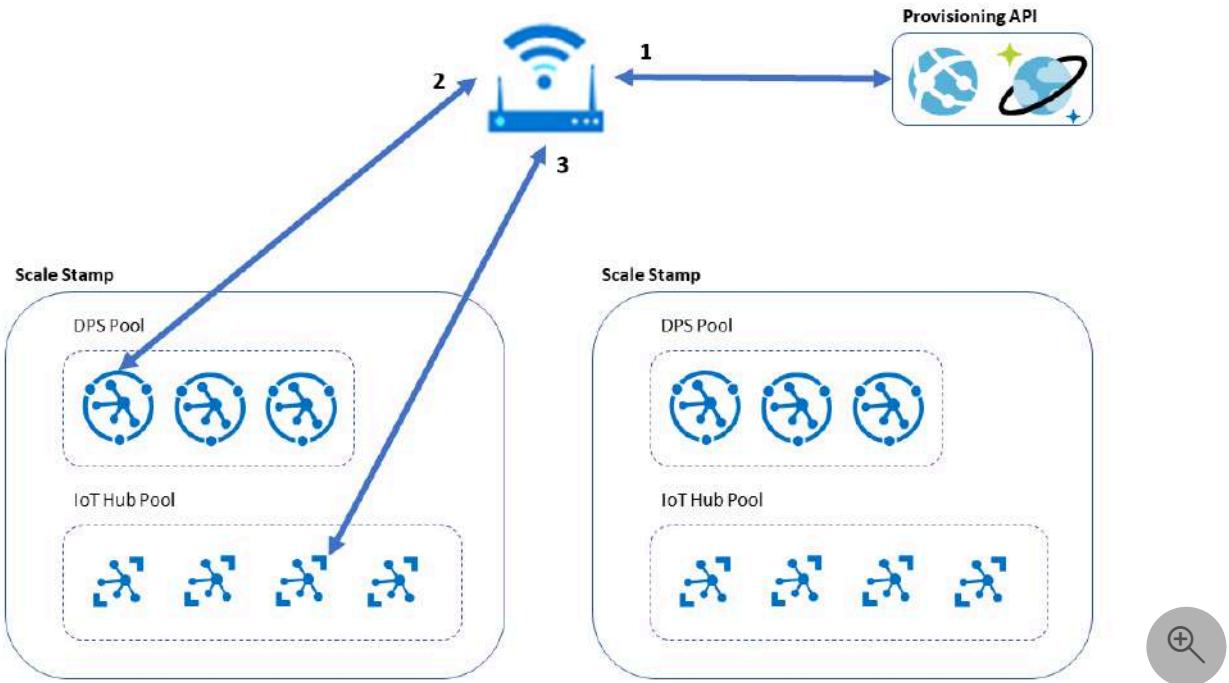
Because of DPS service limits, you often need to expand to multiple DPS instances. You can approach device provisioning across multiple DPS instances through either zero-touch provisioning or low-touch provisioning.

The following approaches apply the previously described *stamp* concept for resiliency and scaling out. This concept includes deploying Azure App Service in multiple regions and using a tool such as [Traffic Manager](#) or a [global load balancer](#). For simplicity, the following diagrams don't show these components.

Approach 1: Zero-touch provisioning with multiple DPS instances

For zero-touch or automated provisioning, a proven strategy includes having the device request a DPS ID scope from a web API. The API understands and balances devices across the horizontally scaled-out DPS instances. This action makes the web app a critical part of the provisioning process, so it must be scalable and highly available. This design has three primary variations.

The following diagram shows the first option that uses a custom provisioning API that manages how to map the device to the appropriate DPS pool. Each DPS instance then maps the device to the appropriate IoT Hub by using standard DPS [load balancing mechanisms](#).



1. The device makes a request to a provisioning API hosted in App Service to request a DPS ID scope. The provisioning API checks with its persistent database to determine the best instance for the device, based on existing device inventory, and returns the DPS ID scope.

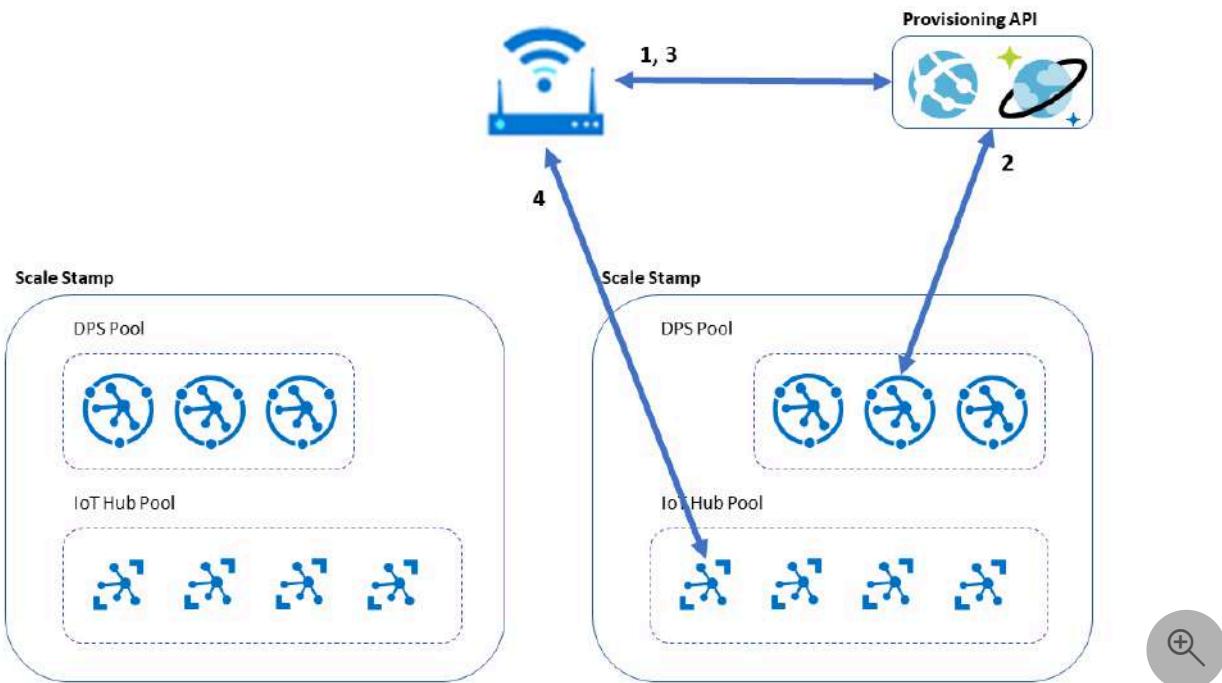
In this example, the database is an Azure Cosmos DB instance that has multi-primary write enabled for cross-region high availability. This database stores each device's assigned DPS. It supports tracking DPS instance usage for all appropriate metrics, such as provision requests per minute and total provisioned devices. This database also enables reprovisioning with the same DPS ID scope when needed. Authenticate the provisioning API to prevent inappropriate provisioning requests.

2. The device makes a request against DPS by using the assigned ID scope. DPS responds with IoT hub assignment details.
3. The device stores the ID scope and IoT hub connection information in persistent storage, ideally in a secured storage location because the ID scope is part of the authentication against the DPS instance. The device then uses this IoT hub connection information for further requests into the system.

This design requires the device software to include the DPS SDK and manage the DPS enrollment process, which is the typical design for an Azure IoT device. But in a microcontroller environment, where device software size is a critical component of the design, it might not be acceptable and might require an alternative design.

Approach 2: Zero-touch provisioning with a provisioning API

The second design moves the DPS call to the provisioning API. In this model, the device authentication against DPS is contained in the provisioning API, as is most of the retry logic. This process allows more advanced queuing scenarios and potentially simpler provisioning code in the device itself. It also allows for caching the assigned IoT hub to facilitate faster C2D messaging. The messages are sent without needing to interrogate DPS for the assigned hub information.



1. The device makes a request to a provisioning API hosted in an instance of App Service. The provisioning API checks with its persistent database to determine the best instance for the device based on existing device inventory, and then it determines the DPS ID scope.

In this example, the database is an Azure Cosmos DB instance that has multi-primary write enabled for cross-region high availability. This database stores each device's assigned DPS. It supports tracking DPS instance usage for all appropriate metrics. The database also enables reprovisioning by using the same DPS ID scope when needed.

Authenticate the provisioning API to prevent inappropriate provisioning requests. You can likely use the same authentication that the provisioning service uses against DPS, such as a private key for an issued certificate. But other options exist. For example, FastTrack for Azure might work with a customer that uses hardware unique identifiers as part of their service authentication process. The device manufacturing partner regularly supplies a list of unique identifiers to the device vendor to load into a database, which references the service behind the custom provisioning API.

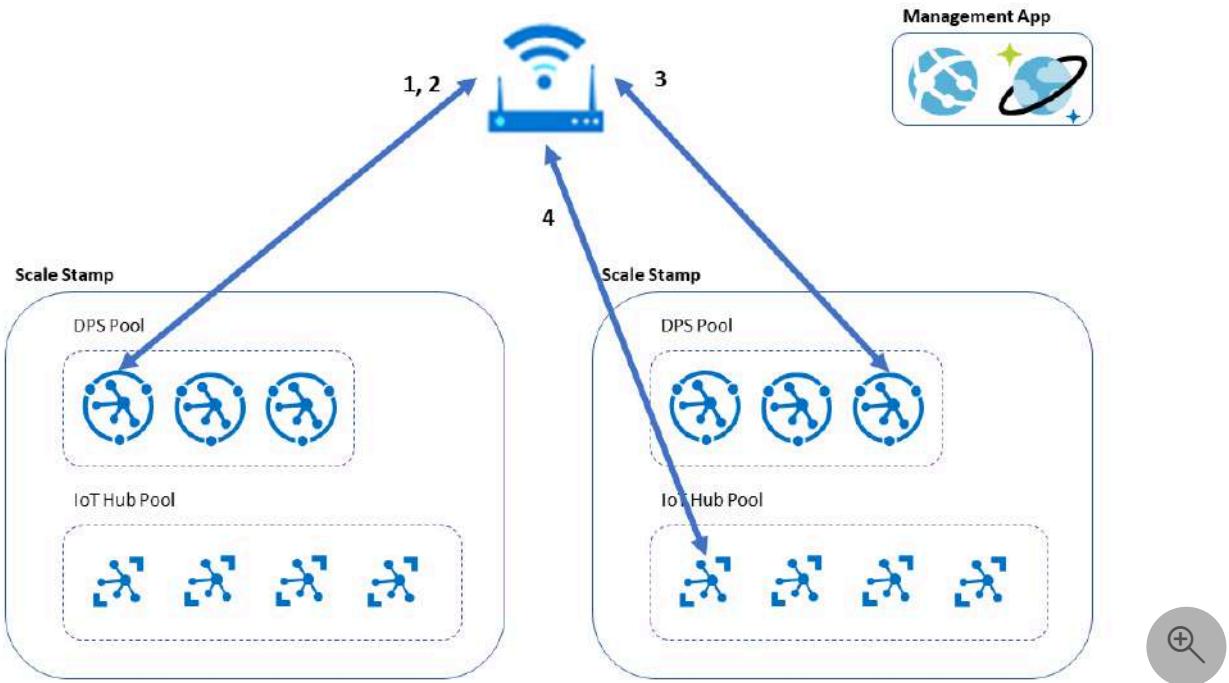
2. The provisioning API performs the DPS provisioning process by using the assigned ID scope, effectively acting as a DPS proxy.
3. The API forwards the DPS results to the device.
4. The device stores the IoT hub connection information in persistent storage, ideally in a secured storage location because the ID scope is part of the authentication against the DPS instance. The device uses this IoT hub connection information for later requests into the system.

This design avoids the need to reference the DPS SDK or the DPS service. It also avoids the need for storing or maintaining a DPS scope on the device. This model supports transfer-of-ownership scenarios because the provisioning service can direct to the appropriate final customer DPS instance. However, this approach causes the provisioning API to duplicate some DPS functionality, which might not suit all scenarios.

Approach 3: Zero-touch provisioning with transfer of ownership

A third zero-touch provisioning design uses a factory-configured DPS instance as a starting point and redirects devices to other DPS instances as necessary. This design allows for provisioning without a custom provisioning API, but it requires a management application to track DPS instances and supply redirection as necessary.

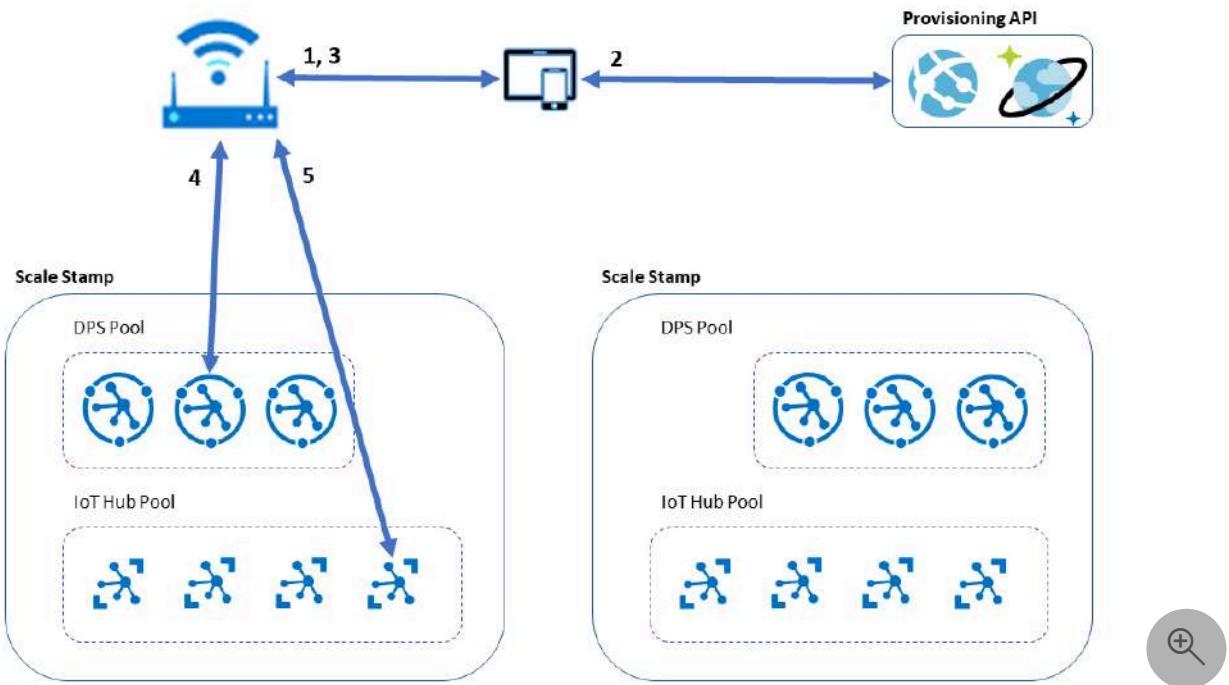
The management application requirements include tracking which DPS should be the active DPS for each specific device. You can use this approach for transfer-of-ownership scenarios, where the device vendor transfers ownership of the device from the vendor to the end device customer.



1. The device connects to the factory-configured DPS instance and requests an initial provisioning process.
2. The device receives an initial configuration, including the desired target DPS instance.
3. The device connects to the desired target DPS instance and requests provisioning.
4. The device stores the IoT hub connection information in persistent storage, ideally in a secured storage location because the ID scope is part of the authentication against the DPS instance. The device uses this IoT hub connection information for further requests into the system.

Approach 4: Low-touch provisioning with multiple DPS instances

In some cases, such as in consumer-facing scenarios or field deployment team devices, a common choice is to offer low-touch, or user-assisted, provisioning. Examples of low-touch provisioning include a mobile application on an installer's phone or a web-based application on a device gateway. This approach performs the same operations as the zero-touch provisioning process, but the provisioning application transfers the details to the device.



1. The administrator launches a device configuration app, which connects to the device.
2. The configuration app connects to a provisioning API hosted in an instance of App Service to request a DPS ID scope. The provisioning API checks its persistent database to determine the best instance for the device, based on existing device inventory, and returns the DPS ID scope.

In this example, the database is an Azure Cosmos DB instance that has multi-primary write enabled for cross-region high availability. This database stores each device's assigned DPS. It supports tracking usage of the DPS instances for all appropriate metrics. This database also enables reprovisioning by using the same DPS ID scope when needed. Authenticate the provisioning API to prevent inappropriate provisioning requests from being serviced.

3. The app returns the provisioning ID scope to the device.
4. The device makes a request against DPS with the assigned ID scope. DPS returns IoT hub assignment details to the device.
5. The device persists the ID scope and IoT hub connection information to persistent storage, ideally in a secured storage location because the ID scope is part of the authentication against the DPS instance. The device uses this IoT hub connection information for further requests into the system.

This article doesn't cover other variations. For example, you can configure this approach by moving the DPS call to the provisioning API, as shown earlier in the zero-touch provisioning

with a provisioning API. The goal is to make sure that each tier is scalable, configurable, and readily deployable.

General DPS provisioning guidance

Apply the following recommendations to your DPS deployment:

Don't provision on every boot. The [DPS documentation](#) recommends that you don't provision on every boot. For small use cases, it might seem reasonable to provision on every boot because that's the shortest path to deployment. However, when you scale up to millions of devices, DPS can become a bottleneck because of its [hard limit of 1,000 registrations per minute, per service instance](#). Even device registration status lookups can become a bottleneck because they have a limit of 5 to 10 polling operations per second. Provisioning results typically map statically to an IoT hub. So you should only initiate provisioning when necessary, unless your requirements include automated reprovisioning requests. If you anticipate more traffic, scale out to multiple DPS instances to support your scenario.

Use a staggered provisioning schedule. To reduce time-based limitations, use a [staggered provisioning schedule](#). For initial provisioning, you can apply a random delay of a few seconds or extend the delay to minutes, depending on the deployment requirements.

Always query status before requesting provisioning. As a best practice, devices should always query their status before requesting provisioning by using the [Device Registration Status Lookup API](#). This call doesn't count as a billed item, and the [limit is independent of the registration limit](#). The query operation is relatively quick compared to a provision request, which means that the device can validate its status and move on to the normal device workload more quickly. For the appropriate device registration logic, see [Large-scale deployment](#).

Follow provisioning API considerations. Several of the designs in this article include a provisioning API. The provisioning API needs a backing metadata store, such as Azure Cosmos DB. At these scale levels, you should implement a globally available and resilient design pattern. The built-in multi-primary, geo-redundant capabilities and latency guarantees in Azure Cosmos DB make it an excellent choice for this scenario. This API has the following key responsibilities:

- **Serve the DPS ID scope.** This interface might use a GET request. Physical devices or management applications connect to this interface.
- **Support the device life cycle.** A device might require reprovisioning, or unexpected events can happen. At a minimum, maintain the device ID and assigned DPS for a device. This information enables deprovisioning from the assigned DPS and reprovisioning on another. Or if a device's life cycle is over, you can completely remove it from the system.

- **Load balance systems.** The system uses the same metadata regarding device ID and DPS, so it can understand the current load on each subsystem and apply that information to better balance devices across the horizontally scaled-out components.
- **Uphold system security.** The provisioning API should authenticate each request. The recommended best practice is to [use a unique X.509 certificate for each device](#). This certificate can authenticate the device against both the provisioning API and the DPS instance, if the architecture supports it. Other methods, such as fleet certificates and tokens, are available but provide less security. Your specific implementation and its security implications depend on whether you choose a zero-touch or low-touch option.

Scale out IoT Hub

Compared to scaling out DPS, scaling out IoT Hub is relatively straightforward. One of the benefits of DPS is its ability to link to many IoT Hub instances. When you follow the recommended practice of using DPS in Azure IoT solutions, scaling out IoT Hub involves the following steps:

1. Create a new instance of the IoT Hub service.
2. Configure the new instance with appropriate routing rules and other details.
3. Link the new instance to the appropriate DPS instances.
4. If necessary, reconfigure the [DPS allocation policy](#) or your [custom allocation policy](#).

Design device software

Scalable device design requires following best practices and device-side considerations. Some of these practices come from anti-patterns encountered in the field. This section describes key concepts for a successfully scaled deployment.

Estimate workloads across different parts of the device life cycle and scenarios within the life cycle. Device registration workloads can vary greatly between development phases, such as pilot, development, production, decommissioning, and end of life. In some cases, they can also vary based on external factors such as the previously mentioned blackout scenario. Design for the worst-case workload to help ensure success at scale.

Support reprovisioning on demand. You can provide this feature through a device command and an administrative user request. For more information, see [Reprovision devices](#). This option facilitates transfer-of-ownership scenarios and factory-default scenarios.

Avoid unnecessary reprovisioning. Active, working devices rarely require reprovisioning because provisioning information remains relatively static. Don't reprovision without a good reason.

Check provisioning status if you must reprovision often, for example at every device boot. If you're unsure about the device provisioning status, [query the provisioning status](#) first. The query operation is handled against a different quota than a provisioning operation and is faster. This query allows the device to validate the provisioning status before proceeding. This approach is especially useful when a device doesn't have available persistent storage to store the provisioning results.

Ensure a good retry logic strategy. The device must have appropriate retry algorithms built into the device code for both initial provisioning and later reprovisioning. Use techniques such as *exponential backoff with randomization*. Initial provisioning, by definition, might need to be more aggressive in the retry process than reprovisioning, depending on the use case. When throttled, DPS returns an [HTTP 429 Too Many Requests](#) error code, like most Azure services. For more information, see [Retry](#) and [Anti-patterns to avoid](#). The DPS documentation also explains how to interpret service retry recommendations and calculate jitter. The device location stability and connectivity access also influence the appropriate retry strategy. For example, if a device detects that it's offline for periods of time, it should avoid retrying online operations.

Support over-the-air (OTA) updates. Two simple update models include using device twin properties with [automatic device management](#) and using simple device commands. For more sophisticated update scenarios and reporting, see [Azure Device Update](#). OTA updates allow you to fix defects in device code and reconfigure services, such as DPS ID scope, if necessary.

Architect for certificate changes at all layers and certificate uses. This recommendation aligns with the OTA update best practices. You must consider certificate rotation. The IoT Hub DPS documentation addresses this scenario [from a device identity certificate](#) viewpoint. In a device solution, other certificates are used for access to services like IoT Hub, App Service, and Azure Storage accounts. Azure sometimes changes certificate authority configurations, so you must anticipate changes at all layers. Also, use certificate pinning with caution, especially when certificates are outside the device manufacturer's control.

Consider a reasonable default state. To resolve initial provisioning failures, have a reasonable disconnected or unprovisioned configuration, depending on the circumstances. If the device has a heavy interaction component as part of initial provisioning, the provision process can occur in the background while the user performs other provisioning tasks. Always pair this approach with an appropriate retry pattern and the [Circuit Breaker pattern](#).

Include endpoint configuration capabilities where appropriate. Allow configuration of the DPS ID scope, the DPS endpoint, or the custom provisioning service endpoint. Although the

DPS endpoint rarely changes, enabling this flexibility supports scenarios such as automated validation of the device provisioning process through integration testing without direct Azure access or future provisioning models that use a proxy service.

Use the Azure IoT SDKs for provisioning. Whether the DPS calls are on the device itself or in a custom provisioning API, use the Azure IoT SDKs to benefit from built-in best practices and simplify support. The SDKs are published open source, so you can review how they work and suggest changes. Your choice of SDK depends on the device hardware and available runtimes on the device.

Deploy devices

Device deployment is a key part of the device life cycle, but it's outside the scope of this article because it's dependent on the use case. The previously referenced discussion points about transfer of ownership might apply to the deployment and patterns that involve a provisioning application, such as a mobile application. But you must select the deployment approach based on the IoT device type in use.

Monitor devices

An important part of your overall deployment is to monitor the solution from start to finish to make sure that the system performs appropriately. This article explicitly focuses on architecture and design and not the operational aspects of the solution, so monitoring isn't included. However, at a high level, monitoring tools are built into Azure through [Azure Monitor](#) to ensure that the solution doesn't reach limits. For more information, see the following articles:

- [Monitor IoT Hub](#)
- [Diagnose and troubleshoot disconnects by using IoT Hub DPS](#)

You can use these tools individually or as part of a more sophisticated Security Information and Event Management (SIEM) solution like [Microsoft Sentinel](#).

Use the following [monitoring patterns](#) to monitor DPS usage over time:

- Create an application that queries each enrollment group on a DPS instance, retrieves the total devices registered to that group, and then aggregates the numbers from across various enrollment groups. This method provides an exact count of devices currently registered via DPS and helps monitor the state of the service.
- Monitor device registrations over a specific period. For instance, you can monitor registration rates for a DPS instance over the previous five days. This approach provides only an approximate figure and is limited to a set time period.

Conclusion

Scaling up an IoT solution to support millions, or even hundreds of millions, of devices requires careful planning. You must consider many factors and various ways to solve the problems that arise at those scales. This article summarizes the key concerns and provides approaches about how to address those concerns in a successful deployment.

Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal author:

- [Michael C. Bazarewsky](#) | Senior Customer Engineer, Microsoft Azure CXP

Other contributors:

- [David Crook](#) | Principal Customer Engineering Manager, Microsoft Azure CXP
- [Alberto Gorni](#) | Former Senior Customer Engineer, Microsoft Azure CXP

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

- [Best practices for large-scale Microsoft Azure IoT device deployments](#)
- [Protect your cloud estate](#)

Related resources

- [Move an IoT solution from test to production](#)
- [Transient fault handling](#)

Move an IoT Hub-based solution from test to production

Azure IoT Hub

This article provides a list of key considerations for transitioning an Azure IoT Hub-based solution to a production environment.

Use deployment stamps

Deployment stamps are discrete units of core solution components that support a defined number of devices. A single unit is known as a *stamp* or *scale unit*. A stamp might consist of a set device population, an IoT hub, an event hub or other routing endpoint, and a processing component. Each stamp supports a defined device population. You choose the maximum number of devices that the stamp can hold. As the device population grows, you add stamp instances instead of independently scaling up different parts of the solution.

If you move a single instance of your IoT Hub-based solution to production instead of adding stamps, you might encounter the following limitations:

- **Scale limits.** Your single instance can experience scaling limits. For example, your solution might use services that have limits on the number of inbound connections, host names, Transmission Control Protocol sockets, or other resources.
- **Non-linear scaling or cost.** Your solution components might not scale linearly with the number of requests or the volume of ingested data. Instead, some components might experience performance degradation or increased costs after a threshold is reached. In such cases, scaling out by adding stamps might be a more effective strategy than increasing capacity.
- **Separation of customers.** You might need to isolate specific customers' data from other customers' data. You can group customers that have higher resource demands on different stamps.
- **Single and multitenant instances.** You might have several large customers who need their own independent instances of your solution. Or you might have a pool of smaller customers who can share a multitenant deployment.
- **Complex deployment requirements.** You might need to deploy updates to your service in a controlled manner and deploy to different stamps at different times.

- **Update frequency.** You might have customers who are tolerant of frequent system updates, while other customers might be risk-averse and want infrequent updates to your service.
- **Geographical or geopolitical restrictions.** To reduce latency or comply with data sovereignty requirements, you can deploy some of your customers into specific regions.

To avoid the previous problems, consider grouping your service into multiple stamps. Stamps operate independently of each other and can be deployed and updated independently. A single geographical region might contain a single stamp or might contain multiple stamps to enable horizontal scale-out within the region. Each stamp contains a subset of your customers.

For more information, see [Deployment Stamps pattern](#).

Use back-off when a transient fault occurs

All applications that communicate with remote services and resources must be designed to handle transient faults. This need is especially crucial in cloud environments, where connectivity increases the likelihood of encountering these faults. Transient faults include:

- Momentary loss of network connectivity to components and services.
- Temporary unavailability of a service.
- Time-outs that occur when a service is busy.
- Collisions that occur when devices transmit simultaneously.

These faults are often self-correcting, and if the action is repeated after a suitable delay, it's likely to succeed. However, determining the appropriate intervals between retries is difficult. Typical strategies use the following types of retry intervals:

- **Exponential back-off.** The application waits a short time before the first retry, then progressively increases the wait time between subsequent attempts. For example, it might retry the operation after 3 seconds, 12 seconds, or 30 seconds.
- **Regular intervals.** The application waits for the same period of time between each attempt. For example, it might retry the operation every 3 seconds.
- **Immediate retry.** Sometimes a transient fault is brief and can occur because of events like a network packet collision or a spike in a hardware component. In this scenario, retrying the operation immediately is appropriate. If the fault clears by the time the application assembles and sends the next request, the operation can succeed. However, there should never be more than one immediate retry attempt. If the immediate retry fails, switch to alternative strategies, such as exponential back-off or fallback actions.

- **Randomization.** Any of the previous retry strategies might include a randomization element to prevent multiple instances of the client from sending subsequent retry attempts at the same time.

Avoid the following anti-patterns:

- Don't include duplicated layers of retry code in implementations.
- Never implement an endless retry mechanism.
- Never perform an immediate retry more than one time.
- Avoid using a regular retry interval.
- Prevent multiple instances of the same client, or multiple instances of different clients, from sending retries at the same time.

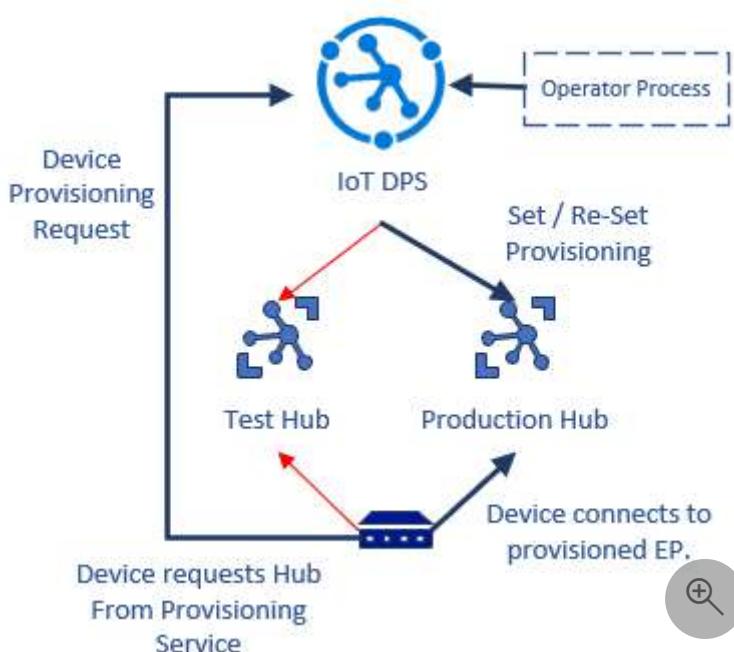
For more information, see [Transient fault handling](#).

Use zero-touch provisioning

Provisioning is the process of enrolling a device into IoT Hub. Provisioning registers a device with IoT Hub and specifies the attestation mechanism that it uses. You can use the [IoT Hub device provisioning service \(DPS\)](#) or provision directly via IoT Hub Registry Manager APIs.

Using DPS provides the advantage of late binding, which allows the removal and reprovisioning of field devices to IoT Hub without changes to the device software.

The following example shows how to implement a test-to-production environment transition workflow by using DPS.



1. The solution developer links both the test and production IoT clouds to the provisioning service.
2. The device implements the DPS protocol to find the IoT hub, if it's no longer provisioned. The device is initially provisioned to the test environment.
3. The device is registered with the test environment, so it connects there and testing occurs.
4. The developer reprovisions the device to the production environment and removes it from the test hub. The test hub rejects the device the next time that it reconnects.
5. The device connects and renegotiates the provisioning flow. DPS directs the device to the production environment and the device connects and authenticates there.

For more information, see [Overview of IoT Hub device provisioning service](#).

Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal authors:

- [Matthew Cosner](#) | Principal Software Engineering Manager
- [Ansley Yeo](#) | Principal Program Manager

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next step

- [Browse Azure architectures](#)

Architectural approaches for IoT Hub-based multitenant solutions

Article • 12/13/2024

Multitenant IoT Hub-based solutions come in many different flavors and sizes. You might have many requirements and constraints, ranging from infrastructure ownership, to customer data isolation, to compliance. It can be challenging to define a pattern that meets all of these design constraints, and doing so often requires considering multiple dimensions. This article describes several approaches commonly used to solve multitenancy considerations for IoT Hub-based solutions.

Key considerations and requirements

These considerations and requirements are presented in the order in which they're typically prioritized for a solution's design.

Governance and compliance

Governance and compliance considerations might require that you use a particular pattern or set of IoT resources. Not all IoT services have the same certifications or capabilities. If you need to meet specific compliance standards, you might need to select specific services. To learn more, see [Architectural approaches for governance and compliance in multitenant solutions](#).

Governance in IoT can also take other forms, such as device ownership and management. Does the customer own the device or does the solution provider? Who owns the management of those devices? These considerations and implications are unique to each solution provider and can lead to different choices in the technology, deployment pattern, and multitenancy pattern that you use.

Scale

It's important to plan your solution's scale. Scale is often considered across these three dimensions:

- **Quantity of devices:** All Azure device management services - [Azure IoT Central](#), [Azure IoT Hub Device Provisioning Service \(DPS\)](#), and [Azure IoT Hub](#) - have limitations on the number of devices supported in a single instance.

💡 Tip

Refer to the [high scale documentation](#), if you plan to deploy a very large number of devices.

- **Device throughput:** Different devices, even in the same solution, might have different throughput requirements. "Throughput" in this context refers to both the number of messages over a period of time and the size of the messages. For example, in a:
 - Smart-building solution, thermostats typically report data at a lower frequency than elevators.
 - In a connected-vehicle solution, vehicle camera recording data messages are typically larger than navigation telemetry messages.

If your messages are throttled with respect to frequency, consider scaling out to more instances of a service. If your messages are throttled with respect to size, consider scaling up to larger instances of a service.

- **Tenants:** A single tenant's scale can be small, but when multiplied by the number of tenants, it can quickly grow.

Performance and reliability

Tenant isolation

Fully shared solutions can have [noisy neighbors](#). In the cases of IoT Hub and IoT Central, noisy neighbors can result in HTTP 429 ("Too Many Requests") response codes, which are hard failures that can cause a cascading effect. For more information, see [Quotas and Throttling](#).

In fully multitenant solutions, these effects can cascade. When customers share IoT Hub or IoT Central applications, then all customers on the shared infrastructure receive errors. Because IoT Hub and IoT Central are commonly the entry points for data to the cloud, other downstream systems that depend on this data are likely to fail as well. Often, the most common reason for these errors is when a message quota limit is exceeded. In this situation, the fastest and simplest fix for IoT Hub solutions is to upgrade the IoT Hub SKU, increase the number of IoT Hub units, or both. For IoT Central solutions, the solution automatically scales as necessary, up to the [documented number of messages supported](#).

You can isolate and distribute tenants across the IoT control, management, and communications planes by using DPS [custom allocation policies](#). Further, when you follow the guidance for [high-scale IoT solutions](#), you can manage other allocation distributions at the DPS load-balancer level.

Data storage, query, usage, and retention

IoT solutions tend to be data-intensive, both when streaming and at rest. For more information on managing data in multitenant solutions, see [Architectural approaches for storage and data in multitenant solutions](#).

Security

IoT solutions often have security considerations at multiple layers, especially in solutions that are deployed in a cloud-modified [Purdue Enterprise Reference Architecture](#) or [Industry 4.0](#) solutions. The design approach you select affects what network layers and boundaries exist; after you select the physical design, you can select the security implementation. You can use the following tools in any approach:

- [Microsoft Defender for IoT](#), a comprehensive IoT monitoring solution you should consider that offers a [per-device IoT license](#) and [OT site licenses](#) for each customer device and/or site. Depending on the approach selected from later in this article, the Microsoft 365 named user licensing scenario might not be possible. In that case, the per-device and site license options are available, which are license options independent of Microsoft 365 tenant licenses.
- [Azure Firewall](#) or a non-Microsoft firewall appliance, which you should consider for isolating the network layers and monitoring network traffic. The exact choice of approach determines where workloads have network isolation versus a shared network, as addressed later in this article.
- [Azure IoT Edge](#).

Most of these security topics apply in a multitenant solution similar to how they would in a single tenant solution, with the variations tied to the selected approach. One component that is going to likely be substantially different in an overall IoT solution is user and application identity. [Architectural approaches for identity in multitenant solutions](#) discusses this aspect of an overall multitenant solution.

Approaches to consider

The considerations and choices for primary components, such as management, ingestion, processing, storage, and security, are the same for single and multitenant IoT solutions. The primary difference is how you arrange and utilize the components to support multitenancy. For example, common decision points for:

- Storage might be to choose to use SQL Server or Azure Data Explorer.
- The ingestion and management tier is to choose between IoT Hub and IoT Central.

Most IoT solutions fit within a [root architecture pattern](#), which is a combination of the deployment target, tenancy model, and deployment pattern. The key requirements and considerations described previously determine these factors.

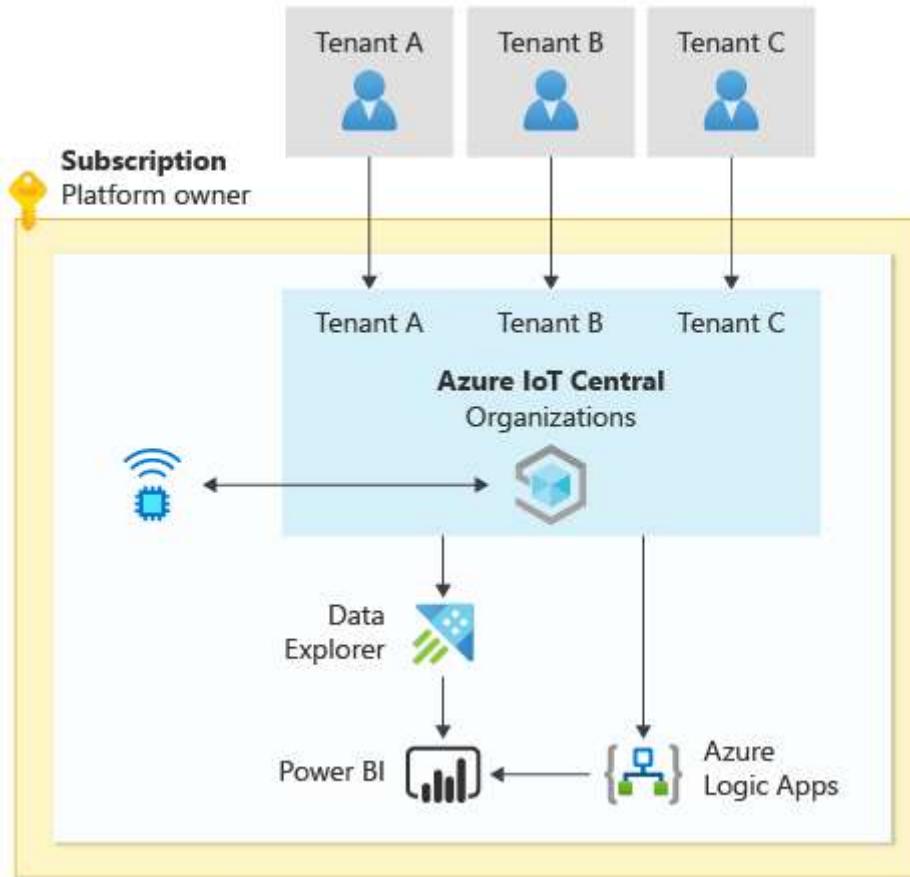
One of the largest decision points needing to be made, within the IoT space, is to select between an application-platform-as-a-service (aPaaS) and platform-as-a-service (PaaS) approaches. To learn more, see [Compare Internet of Things \(IoT\) solution approaches \(PaaS vs. aPaaS\)](#).

This choice is the common "build versus buy" dilemma that many organizations face in many projects. It's important to evaluate the advantages and disadvantages of both options.

Concepts and considerations for aPaaS solutions

A typical aPaaS solution using [Azure IoT Central](#), as the core of the solution, might use the following Azure PaaS and aPaaS services:

- [Azure Event Hubs](#) as a cross-platform, enterprise-grade messaging and data-flow engine.
- [Azure Logic Apps](#) as an integration platform-as-a-service, or iPaaS, offering.
- [Azure Data Explorer](#) as a data analytics platform.
- [Power BI](#) as a visualization and reporting platform.



In the previous diagram, the tenants share an IoT Central environment, Azure Data Explorer, Power BI, and Azure Logic Apps.

This approach is generally the fastest way to get a solution to market. It's a high scale service that supports multitenancy by using [organizations](#).

It's important to understand that because IoT Central is an aPaaS offering, there are certain decisions that are outside of the control of the implementer. These decisions include:

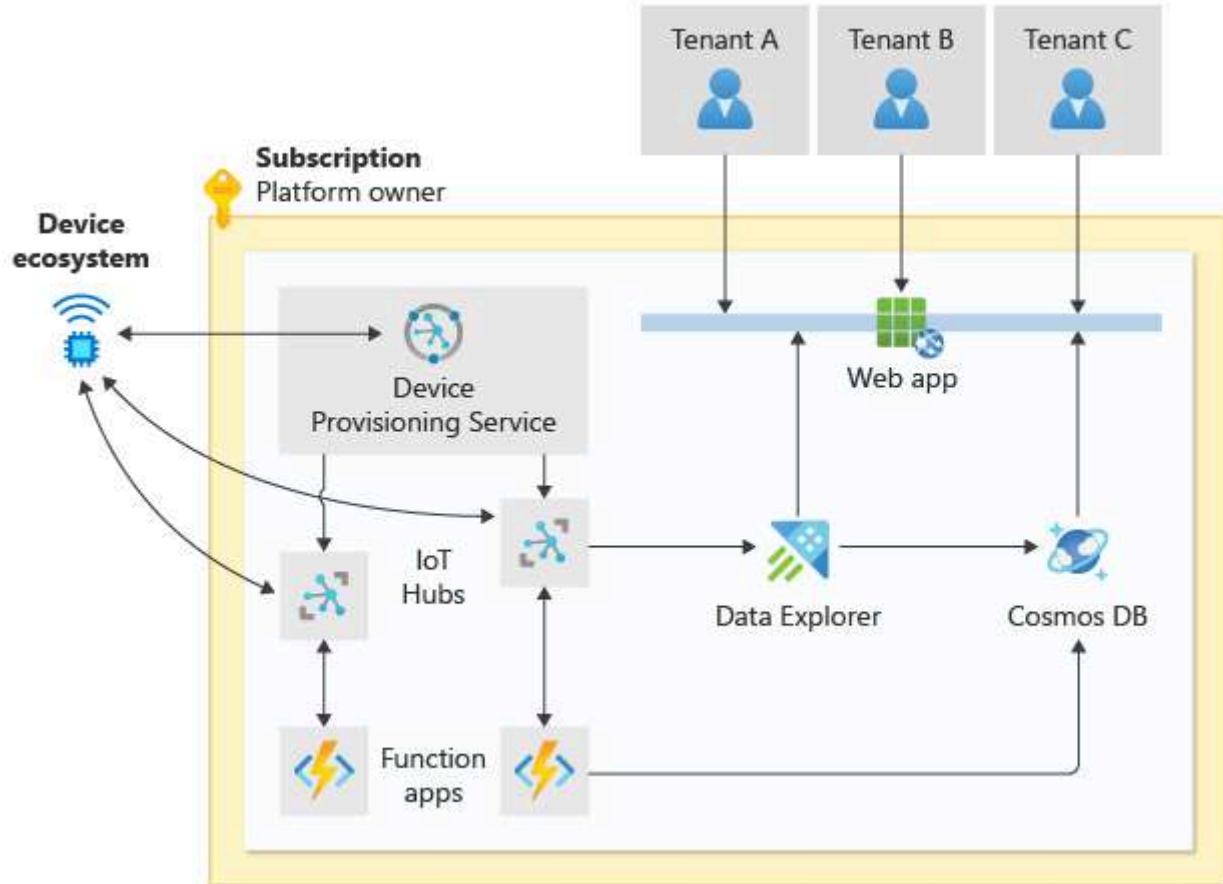
- IoT Central uses Microsoft Entra ID as its identity provider.
- IoT Central deployments are achieved using both control and data plane operations, which combine declarative documents with imperative code.
- [Maximum node limit](#) and maximum tree depth in an IoT Central-based multitenant pattern, might force a service provider to have multiple IoT Central instances. In that case, you should consider following the [Deployment Stamp pattern](#).
- IoT Central imposes [API call limits](#), which might affect large implementations.

Concepts and considerations for PaaS solutions

A PaaS-based approach might use the following Azure services:

- [Azure IoT Hub](#) as the core device configuration and communications platform.

- [Azure IoT Device Provisioning Service](#) as the device deployment and initial configuration platform.
- [Azure Data Explorer](#) for storing and analyzing warm and cold path time series data from IoT devices.
- [Azure Stream Analytics](#) for analyzing hot path data from IoT devices.
- [Azure IoT Edge](#) for running artificial intelligence (AI), non-Microsoft services, or your own business logic on IoT Edge devices.



In the previous diagram, each tenant connects to a shared web app, which receives data from IoT Hubs and a function app. Devices connect to the Device Provisioning Service and to IoT Hubs.

This approach requires more developer effort to create, deploy, and maintain the solution (versus an aPaaS approach). Fewer capabilities are prebuilt for the implementer's convenience. Therefore this approach also offers more control, because fewer assumptions are embedded in the underlying platform.

Root architecture patterns

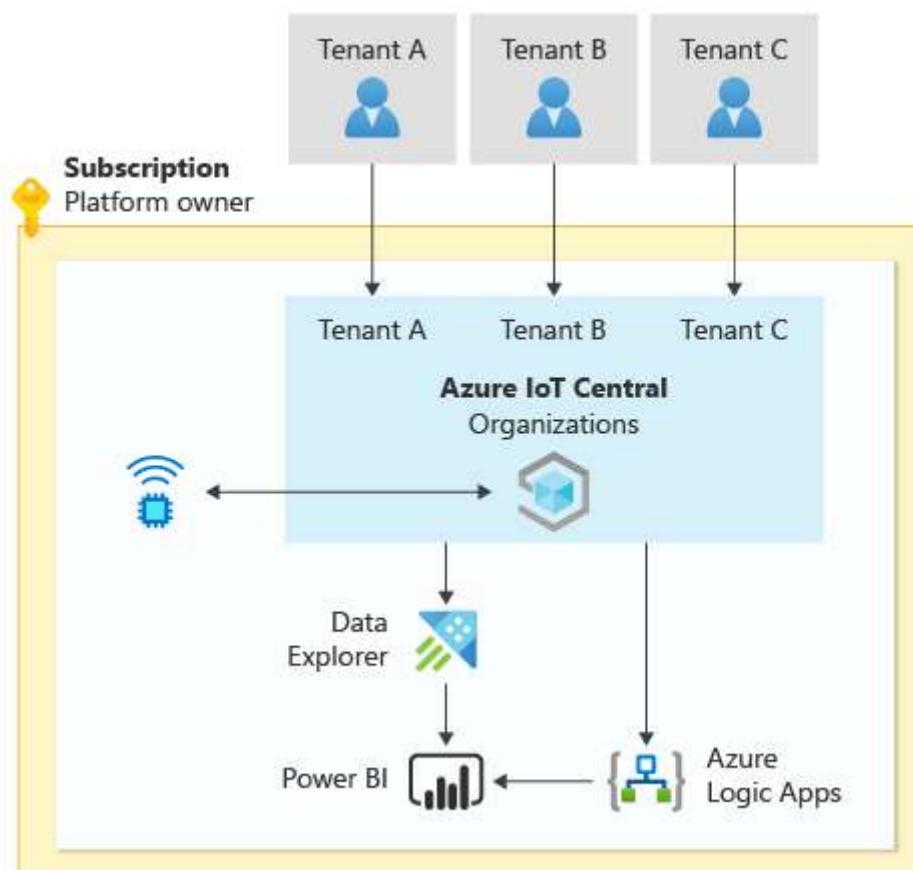
The following table lists common patterns for multitenant IoT solutions. Each pattern includes the following information:

- The name of the **Pattern**, which is based on the combination of the target, model, and deployment type.
- The **Deployment target**, representing the Azure Subscription to deploy resources to.
- The **Tenancy model** being referenced by the pattern, as described at [Multitenancy models](#)
- The **Deployment pattern**, referring to a simple deployment with minimal deployment considerations, a [Geode pattern](#), or a [Deployment Stamp pattern](#).

[Expand table](#)

Pattern	Deployment target	Tenancy model	Deployment pattern
Simple SaaS	Service provider's subscription	Fully multitenant	Simple
Horizontal SaaS	Service provider's subscription	Horizontally partitioned	Deployment Stamp
Single-tenant automated	Either service provider's or customer's subscription	Single tenant per customer	Simple

Simple SaaS



Deployment Target	Tenancy Model	Deployment Pattern
Service provider's subscription	Fully multitenant	Simple

The *Simple SaaS* approach is the simplest implementation for a SaaS IoT Solution. As the previous diagram shows, all of the infrastructure is shared, and the infrastructure has no geographic or scale stamping applied. Often, the infrastructure resides within a single Azure subscription.

[Azure IoT Central](#) supports the concept of [organizations](#). Organizations enable a solution provider to easily segregate tenants in a secure, hierarchical manner, while sharing the basic application design across all the tenants.

Communications to systems outside of IoT Central, such as for longer-term data analysis, along a cold path or connectivity with business operations, is done through other Microsoft PaaS and aPaaS offerings. These other offerings might include the following services:

- [Azure Event Hubs](#) as a cross-platform, enterprise-grade messaging and data flow engine.
- [Azure Logic Apps](#) as an integration platform-as-a-service, or iPaaS.
- [Azure Data Explorer](#) as a data analytics platform.
- [Power BI](#) as a visualization and reporting platform.

If you compare the *Simple SaaS* approach with the [Single tenant automated](#) aPaaS model, many characteristics are similar. The primary difference between the two models is that:

- In the *Single tenant automated* model, you deploy a distinct IoT Central instance for each tenant,
- In the *Simple SaaS with aPaaS* model, you deploy a shared instance for multiple customers, and you create an IoT Central organization for each tenant.

Because you share a multitenant data tier in this model, you need to implement row-level security in order to isolate the customer data. To learn more, see [Architectural approaches for storage and data in multitenant solutions](#).

Benefits:

- Easier to manage and operate relative to the other approaches presented here.

Risks:

- This approach might not easily [scale to high numbers of devices, messages, or tenants](#).
- Services and components are shared, therefore a failure in any component might affect all of your tenants. This risk is a risk to your solution's reliability and high availability.
- It's important to consider how you manage the compliance, operations, tenant lifecycle, and security of subfleets of devices. These considerations become important because of the shared nature of this solution type at the control, management, and communications planes.

Horizontal SaaS

[\[+\] Expand table](#)

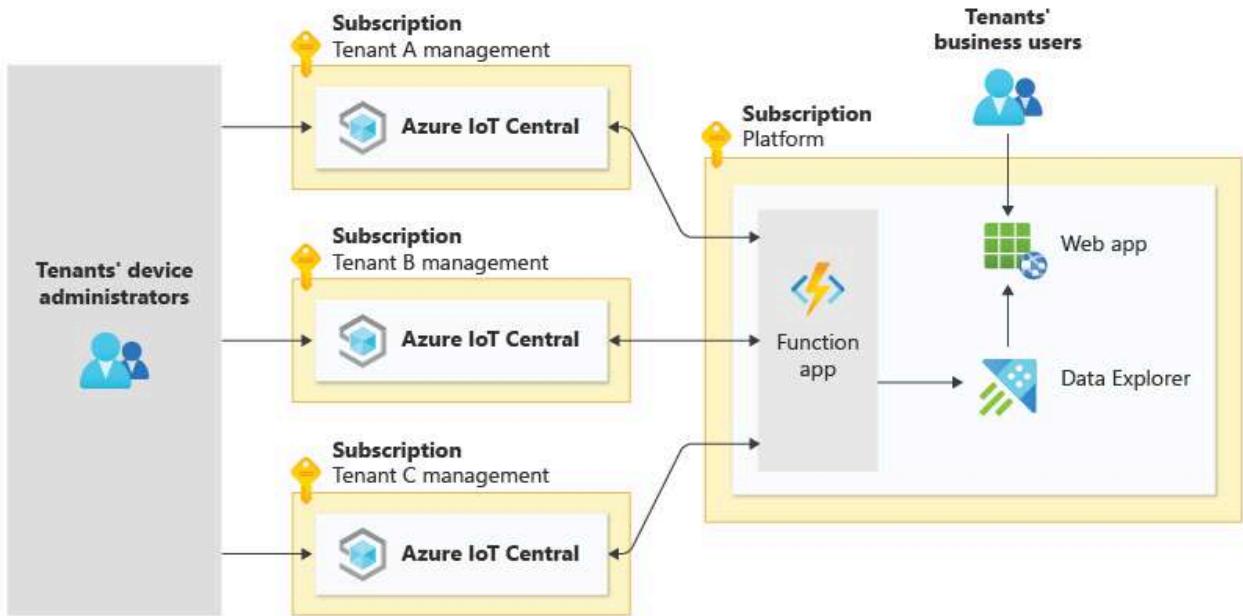
Deployment target	Tenancy model	Deployment pattern
Service provider's subscription	Horizontally partitioned	Deployment Stamp

A common scalability approach is to [horizontally partition the solution](#). This means you have some shared components and some per-customer components.

Within an IoT solution, there are many components that can be horizontally partitioned. The horizontally partitioned subsystems are typically arranged using a [deployment stamp pattern](#) which integrates with the greater solution.

Example horizontal SaaS

The following architectural example partitions IoT Central per end customer, which serves as the device management, device communications, and administrations portal. This partitioning is often done in such a way that the end customer who consumes the solution has full control over adding, removing, and updating their devices, without intervention from the software vendor. The rest of the solution follows a standard shared infrastructure pattern, which solves for hot path analysis, business integrations, SaaS management, and device analysis needs.



Each tenant has their own IoT Central organization, which sends telemetry to a shared function app and makes it available to the tenants' business users through a web app.

Benefits:

- Easy to manage and operate, although extra management might be required for single-tenant components.
- Flexible scaling options, because layers are scaled as necessary.
- Effect of component failures is reduced. While a failure of a shared component affects all customers, horizontally scaled components only affect the customers that are associated with specific scale instances.
- Improved per-tenant consumption insights for partitioned components.
- Partitioned components provide easier per-tenant customizations.

Risks:

- **Scale** of the solution, especially for any shared components.
- Reliability and high availability are potentially affected. A single failure in the shared components might affect all the tenants at once.
- The per-tenant partitioned component customization requires long-term DevOps and management considerations.

Following are the most common components that are typically suitable for horizontal partitioning.

Databases

You might choose to partition the databases. Often it's the telemetry and device data stores that are partitioned. Frequently, multiple data stores are used for different specific

purposes, such as warm versus archival storage, or for tenancy subscription status information.

Separate the databases for each tenant, for the following benefits:

- Support compliance standards. Each tenant's data is isolated across instances of the data store.
- Remediate noisy neighbor issues.

Device management, communications, and administration

Azure IoT Hub Device Provisioning Service, IoT Hub, and IoT Central applications can often be deployed as horizontally partitioned components. In this approach, you need another service to redirect devices to the appropriate DPS instance for that particular tenant's management, control, and telemetry plane. To learn more, see the [Scaling out an Azure IoT solution to support millions of devices](#) whitepaper.

This approach is often taken to enable the end customers to manage and control their own fleets of devices that are more directly and fully isolated.

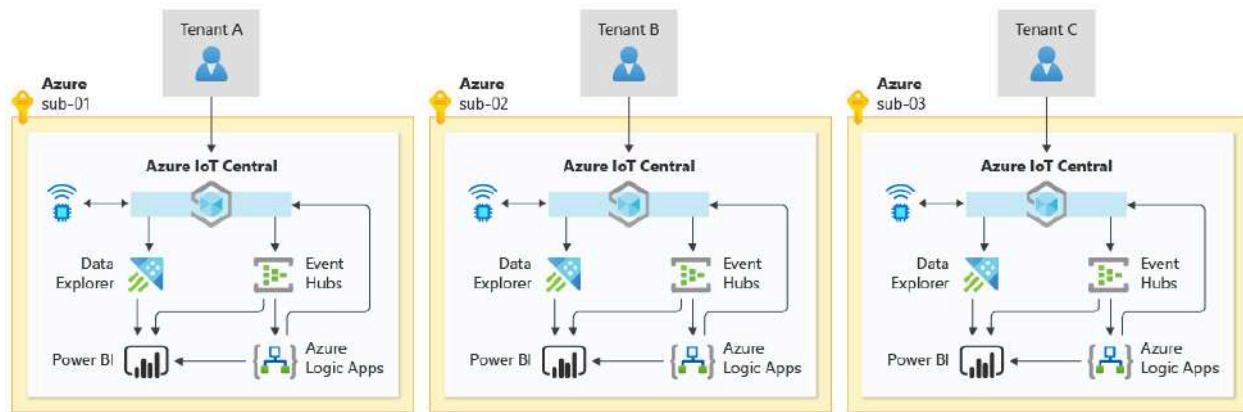
If the device communications plane is horizontally partitioned, telemetry data must be enriched with data identifying the source tenant. This enrichment lets the stream processor know which tenant rules to apply to the data stream. For example, if a telemetry message generates a notification in the stream processor, the stream processor needs to determine the proper notification path for the associated tenant.

Stream processing

By partitioning stream processing, you enable per-tenant customizations of the analysis within the stream processors.

Single-tenant automated

A single-tenant automated approach is based on a similar decision process and design to an [enterprise solution](#).



Each tenant has its own identical, isolated environment, with an IoT Central organization and other components dedicated to them.

[Expand table](#)

Deployment Target	Tenancy Model	Deployment Pattern
Either service provider's or customer's subscription	Single tenant per customer	Simple

A critical decision point in this approach is choosing which Azure subscription the components should be deployed to. If the components are deployed to your subscription, you have more control and better visibility into the cost of the solution, but it requires you to own more of the solution's security and governance concerns. Conversely, if the solution is deployed in your customer's subscription, the customer is ultimately responsible for the security and governance of the deployment.

This pattern supports a high degree of scalability because tenant and subscription requirements are generally the limiting factors in most solutions. Therefore, isolate each tenant to give a large scope for scaling each tenant's workload, without substantial effort on your part, as the solution developer.

This pattern also generally has low latency, when compared to other patterns, because you're able to deploy the solution components based on your customers' geography. Geographical affinity allows for shorter network paths between an IoT device and your Azure deployment.

If necessary, you can extend the automated deployment to support improved latency or scale, by enabling fast deployment of extra instances of the solution in existing or new geographies.

The *single-tenant automated* approach is similar to the *simple SaaS* aPaaS model. The primary difference between the two models is that in the *single-tenant automated*

approach, you deploy a distinct IoT Central instance for each tenant, while in the *simple SaaS* with aPaaS model, you deploy a shared instance of IoT Central with multiple IoT Central organizations.

Benefits:

- Easy to manage and operate.
- Tenant isolation is guaranteed.

Risks:

- Initial automation can be complicated for new development staff.
- Security of cross-customer credentials for higher-level deployment management must be enforced, or the compromises can extend across customers.
- Costs are expected to be higher, because the scale benefits of a shared infrastructure across customers aren't available.
- Many instances to maintain if the solution provider owns the maintenance of each instance.

Increase the scale of SaaS

When you expand the scale of a solution to large deployments, there are specific challenges that arise based on service limits, geographic concerns, and other factors. For more information on large-scale IoT deployment architectures, see [Scaling out an Azure IoT solution to support millions of devices](#).

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal authors:

- [Michael C. Bzarewsky](#) | Senior Customer Engineer, FastTrack for Azure
- [David Crook](#) | Principal Customer Engineer, FastTrack for Azure

Other contributors:

- [John Downs](#) | Principal Software Engineer
- [Arsen Vladimirs](#) | Principal Customer Engineer, FastTrack for Azure

Next steps

- Review guidance for [multitenancy and Azure Cosmos DB](#).
 - Learn about [hot, warm, and cold data paths with IoT on Azure](#).
-

Feedback

Was this page helpful?

 Yes

 No

Data analytics for automotive test fleets

Microsoft Fabric

Azure Data Explorer

Azure Event Hubs

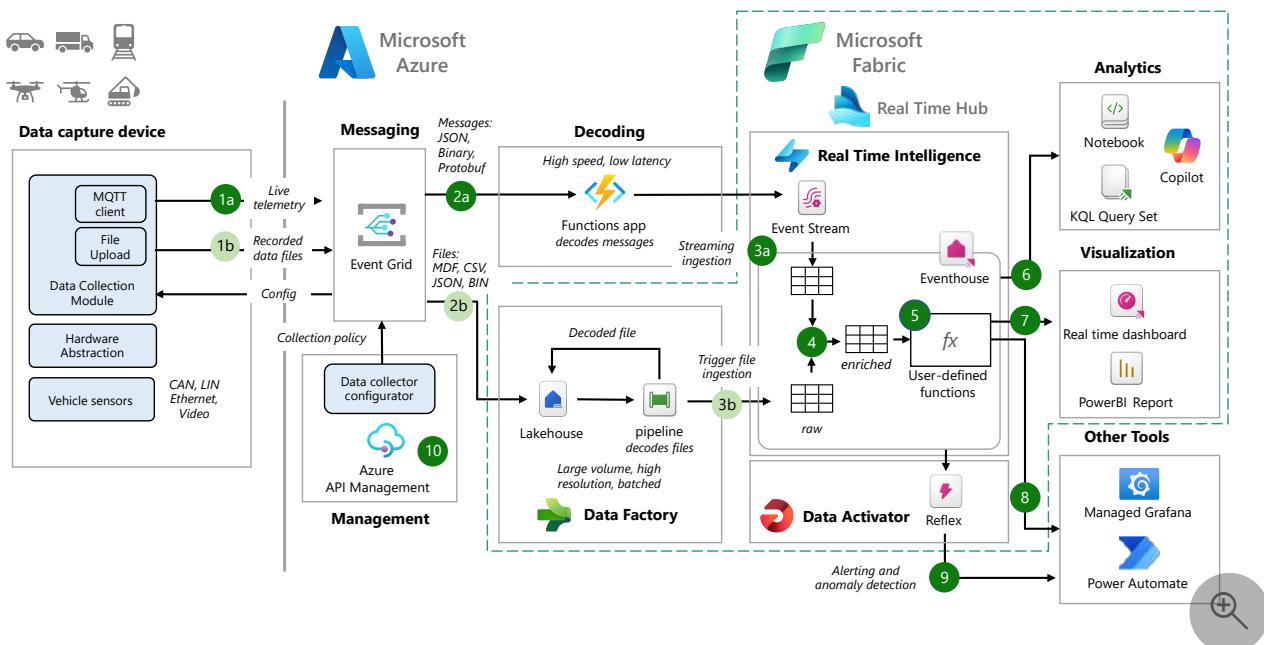
Azure Functions

Azure Event Grid

Automotive original equipment manufacturers (OEMs) need solutions to minimize the time between test drives and delivering test drive diagnostic data to R&D engineers. As vehicles become more automated, the software development lifecycles become shorter, which requires faster digital feedback loops. New technology can democratize data access and provide R&D engineers with near real-time insights into test drive diagnostic data. Use Copilot for Data Science and Data Engineering for data analytics to further reduce the time to insight. Secure data sharing can enhance collaboration between OEMs and suppliers and reduce development cycle times.

The guidance in this article is for telemetry scenarios and batch test drive data ingestion scenarios. This architecture focuses on the data platform that processes diagnostic data and the connectors for data visualization and data reporting.

Architecture



Download a [PowerPoint file](#) with all the diagrams in this article.

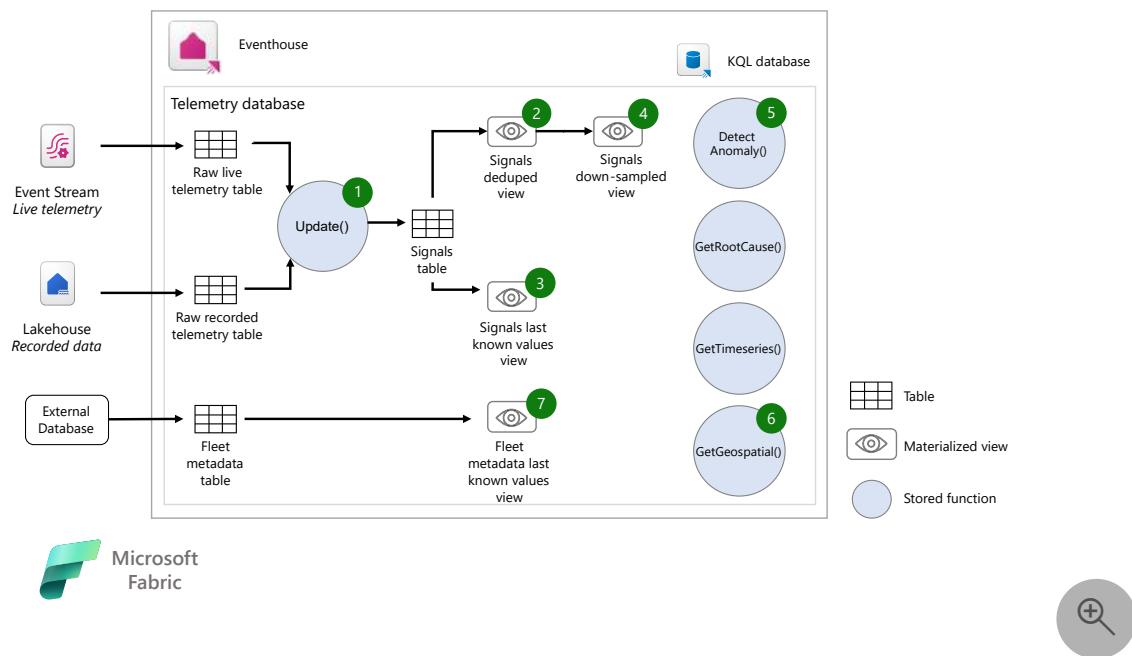
Dataflow

The following dataflow corresponds to the preceding diagram:

1. The data capture device is connected to the vehicle networks and collects high-resolution vehicle signal data and video. (1a) The device publishes real-time telemetry messages or (1b) requests the upload of recorded data files to the Azure Event Grid MQTT broker functionality by using an MQTT client. This functionality uses a Claim-Check pattern.
2. (2a) Event Grid routes live vehicle signal data to an Azure Functions app. This app decodes the vehicle signals to the JavaScript Object Notation (JSON) format and posts them to an eventstream.
(2b) Event Grid coordinates the file upload from the device client to the lakehouse. A completed file upload triggers a pipeline that decodes the data and writes the decoded file to OneLine in a format that's suitable for ingestion, such as parquet or CSV.
3. (3a) The eventstream routes the decoded JSON vehicle signals for ingestion in the Eventhouse.
(3b) A data pipeline triggers the ingestion of decoded files from the lakehouse.
4. The Eventhouse uses [update policies](#) to enrich the data and to expand the JSON data into a suitable row format, for example location data might be clustered to align with geospatial analytics. Every time a new row is ingested, the real-time analytics engine invokes an associated `Update()` function.
5. Data engineers and data scientists use [Kusto Query Language \(KQL\)](#) to build analytics use cases. Users store frequently used cases as shareable user-defined functions. The engineers use built-in KQL functions such as aggregation, time-series analysis, geospatial clustering, windowing, and machine learning plugins with Copilot support.
6. R&D engineers and data scientists use notebooks to analyze data and build test and validation use cases.
 - a. R&D engineers use [KQL query sets](#) and [Copilot for Real-Time Intelligence](#) to perform interactive data analysis.
 - b. Data engineers and data scientists use [notebooks](#) to store and share their analysis processes. With notebooks, engineers can use Azure Spark to run analytics and use Git to [manage the notebook code](#). Users can take advantage of [Copilot for Data Science and Data Engineering](#) to support their workflow with contextual code suggestions.
7. R&D engineers and data scientists can use Power BI with dynamic queries or real-time analytics dashboards to create visualizations to share with business users. These visualizations invoke user-defined functions for ease of maintenance.

8. Engineers can also connect more tools to Microsoft Fabric. For instance, they can connect Azure Managed Grafana to the Eventhouse or create a web application that queries the Eventhouse directly.
9. Data engineers and R&D engineers use [Data Activator](#) to create reflex items to monitor conditions and trigger actions, such as triggering Power Automate flows for business integration. For example, Data Activator can notify a Teams channel if the health of a device degrades.
10. The data collector configuration enables engineers to change the data collection policies of the data capture device. Azure API Management abstracts and secures the partner configuration API and provides observability.

KQL database schema



When [you design the table schema](#), consider the difference between **fact** tables and **dimension** tables. Telemetry is a **fact** table because vehicle signals are progressively appended in a streaming fashion or as part of a complete recording, and telemetry doesn't change. You can classify fleet metadata as a **fact** table that updates slowly.

The vehicle telemetry lands in raw tables. You can use the following message processing concepts to organize the data for analysis and reporting:

- Create update policies to expand the JSON telemetry files into individual vehicle signal records by using methods such as:

- `mv-expand()` expands complex values that are stored in JSON structures into rows with individual signals.
 - `geo_point_to_h3cell()` or `geo_point_to_geohash()` converts latitude and longitude to geohashes for geospatial analytics.
 - `toDouble()` and `toString()` casts extracted values from dynamic JSON objects into the appropriate data types.
 - `lookup` extends the records with values from a dimension table.
- Create a **Signals Deduped** materialized view by using the aggregation function `take_any()` on the unique key and timestamp. This materialized view deduplicates signals.
 - Create a **Signals Last Known Values** materialized view by using the aggregation function `arg_max()` on the timestamp. This materialized view provides an up-to-date status of the vehicles.
 - Create a **Signals Downsampled** materialized view by using the [summarize operator](#) with time bins such as *hourly* and *daily*. This materialized view aggregates signals and simplifies reporting across the fleet.
 - Create user-defined functions that provide anomaly detection or root cause analysis.
 - Use time-series functions for [anomaly detection and forecasting](#) to detect potential problems and predict failures.
 - Use the [scan operator](#) to scan, match, and build sequences from the data. Engineers can use the `scan` operator to detect sequences. For example, if a specific event occurs, then a subsequent event must occur within a certain amount of time.
 - Use machine learning plugins like [autocluster](#) to find common patterns of discrete attributes.
 - Perform geospatial analytics with user-defined functions. Use the [geospatial analytics](#) functions to convert coordinates to a suitable grid system and perform aggregations on the data.
 - Create a **fleet metadata table** to store changes on the vehicle metadata and configuration. Create a **fleet metadata last known values** materialized view to store the latest state of the vehicle fleet based on a last-time modified column.

Components

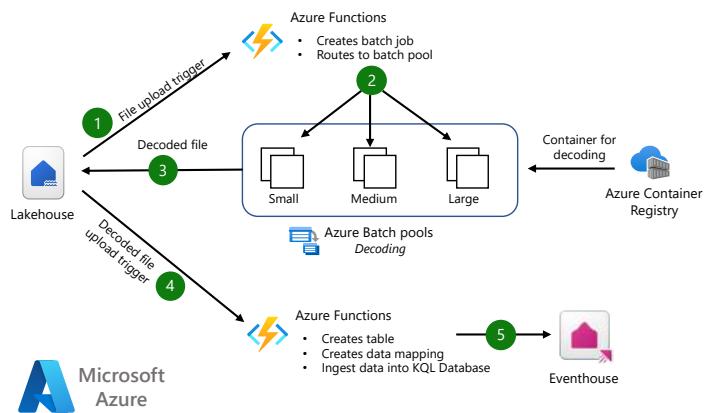
The following key technologies implement this workload. For each component in the architecture, use the relevant service guide in the Well-Architected Framework where available. For more information, see [Well-Architected Framework service guides](#).

- [Fabric Real-Time Intelligence](#) is a service that ingests, analyzes, and reacts to streaming data via eventstreams and KQL databases. In this architecture, it stores and processes vehicle telemetry in motion, which enables real-time analytics and triggers automated actions through reflexes.
- [Data Activator](#) is a no-code automation tool that responds to data patterns and conditions. In this architecture, it monitors telemetry data and triggers actions such as alerts or Power Automate flows when data meets predefined conditions.
- [Event Grid](#) is a managed event routing service that supports MQTT and other protocols. In this architecture, it distributes telemetry and file upload events from vehicles to downstream services like Azure Functions and the lakehouse. Vehicles can use Event Grid to publish and subscribe to topics. For example, they can publish telemetry and subscribe to command and control messages.
- [Azure Event Hubs](#) is a real-time data streaming platform designed for high-throughput scenarios. In this architecture, it ingests millions of vehicle telemetry events per second with low latency for real-time processing.
- [Functions](#) is a serverless compute service that runs code in response to events. In this architecture, it decodes telemetry data, orchestrates file ingestion, and triggers downstream analytics workflows. It simplifies processing vehicle telemetry events at scale with event-driven triggers and bindings by using the language of your choice.
- [Azure Managed Grafana](#) is a data visualization platform based on the software from Grafana Labs. Microsoft manages and supports Azure Managed Grafana. In this architecture, it visualizes telemetry data stored in the eventhouse, which supports dashboards for engineering and operations teams.
- [Azure App Service](#) is a service for building and hosting web apps and APIs. In this architecture, it exposes vehicle telemetry data stored in Fabric through RESTful APIs that external applications consume.
- [API Management](#) is a hybrid multicloud platform for managing APIs. In this architecture, it secures and abstracts access to partner-facing APIs used for configuring data capture devices and integrating business workflows.

Alternatives

You can also use the following Azure services to implement this architecture:

- [Azure Blob Storage](#) stores massive amounts of unstructured data, such as recordings, logs, and videos from the vehicles. It replaces OneLake storage.
- [Azure Data Explorer](#) is a fast, fully managed data analytics service for real-time analysis. It replaces the Fabric Real-Time Intelligence KQL database.
- [Azure Batch](#) is an alternative that you can use to decode complex files. This scenario involves a large number of files that are over 300 megabytes each. The files require different decoding algorithms based on the file version or the file type. You can use either Fabric or use Blob Storage and Azure Data Explorer to implement the following approach.



1. The user or recording device uploads a recorded data file to the lakehouse. When the upload finishes, it triggers a Functions app that schedules decoding.
2. The scheduler starts a Functions app that creates a batch job based on the file type, file size, and required decoding algorithm. The app selects a virtual machine with a suitable size from the pool and starts the job.
3. Batch writes the resulting decoded file back to the lakehouse when the job finishes. This file must be suitable for direct ingestion in a format that the Eventhouse supports.
4. The lakehouse triggers a function that ingests the data into the Eventhouse upon file write. This function creates the table and data mapping if necessary and starts the ingestion process.
5. The KQL database ingests the data files from the lakehouse.

This approach provides the following benefits:

- Functions and Batch pools can handle scalable data processing tasks robustly and efficiently.
- Batch pools provide insight into processing statistics, task queues, and batch pool health. You can visualize status, detect problems, and rerun failed tasks.
- The combination of Functions and Batch supports plug-and-play processing in Docker containers.
- You can use [spot virtual machines](#) to process files during off-peak times. This approach saves money.

Scenario details

Automotive OEMs use large fleets of prototype and test vehicles to test and verify several vehicle functions. Test procedures are expensive because they require real drivers and vehicles, and specific real-world road testing scenarios must pass multiple times. Integration testing is especially important to evaluate interactions between electrical, electronic, and mechanical components in complex systems.

To validate vehicle functions and analyze anomalies and failures, you must capture petabytes of diagnostic data from electronic control units (ECUs), computer nodes, vehicle communication buses like Controller Area Network (CAN) and Ethernet, and sensors.

In the past, small data logger servers in the vehicles stored diagnostic data locally as Measurement Data Format (MDF), multimedia fusion extension (MFX), CSV, or JSON files. After test drives were complete, the servers uploaded diagnostic data to datacenters, which processed it and sent it to R&D engineers for analytics. This process could take hours or sometimes days. More recent scenarios use telemetry ingestion patterns like Message Queuing Telemetry Transport (MQTT)-based synchronous data streams or near real-time file uploads.

Potential use cases

- Vehicle management evaluates the performance and collected data per vehicle across multiple test scenarios.
- System and component validation uses collected vehicle data to verify that the behavior of vehicle components falls within operational boundaries across trips.
- Anomaly detection locates deviation patterns of a sensor value relative to its typical baseline pattern in real time.

- Root cause analysis uses machine learning plugins such as clustering algorithms to identify changes in the distribution of values on multiple dimensions.
- Predictive maintenance combines multiple data sources, enriched location data, and vehicle signals to predict component time to failure.
- Sustainability evaluation uses driver behavior and energy consumption to evaluate the environmental impact of vehicle operations.
- Automotive racing to understand and improve the performance of the vehicles before, during, and after a race.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that can be used to improve the quality of a workload. For more information, see [Microsoft Azure Well-Architected Framework](#).

Reliability

Reliability ensures your application can meet the commitments you make to your customers. For more information, see [Design review checklist for Reliability](#).

- [Azure availability zones](#) ↗ are unique physical locations within the same Azure region. Availability zones can protect Azure Data Explorer compute clusters and data from partial region failure.
- [Business continuity and disaster recovery \(BCDR\)](#) in Azure Data Explorer lets your business continue operating in the face of disruption.
- [Follower databases](#) separate compute resources between production and nonproduction use cases.

Security

Security provides assurances against deliberate attacks and the abuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

It's important to understand the division of responsibility between the automotive OEM and Microsoft. In the vehicle, the OEM owns the whole stack, but as the data moves to the cloud, some responsibilities transfer to Microsoft. Azure platform as a service (PaaS) provides built-in security on the physical stack, including the operating system.

- Use [Azure Policy](#) to apply security guardrails.
- Review the [governance overview and guidance](#) for Fabric.
- Use private endpoints to provide network security for all services.
 - Use [private endpoints for Azure Data Explorer](#).
 - [Allow access to Event Hubs namespaces through private endpoints](#).
- Encrypt data at rest and data in transit.
- Use Microsoft Entra identities and [Microsoft Entra Conditional Access](#) policies.
- Use [row level security \(RLS\)](#) for KQL databases and Azure Data Explorer.
- Use the [restrict statement](#) when you implement middleware applications with access to the KQL database. This configuration creates a logical model that restricts user access to the data.

All these features help automotive OEMs create a secure environment for their vehicle telemetry data. For more information, see [Security in Fabric](#).

Cost Optimization

Cost Optimization is about looking at ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

This solution uses the following practices to help optimize costs:

- Correctly configure hot caches and cold storage for the raw and signals tables. The hot data cache is stored in RAM or SSD and provides improved performance. Cold data, however, is 45 times cheaper. Set a hot cache policy that's adequate for your use case, such as 30 days.
- Set up a retention policy on the raw table and signals table. Determine when the signal data is no longer relevant, such as after 365 days, and set the retention policy accordingly.
- Consider which signals are relevant for analysis.
- Use materialized views when you query the signals last-known values, signals deduped, and signals downsampled. Materialized views consume fewer resources than doing source table aggregations on each query.

- Consider your real-time data analytics needs. Set up streaming ingestion for the live telemetry table to provide latency of less than one second between ingestion and query. This approach increases CPU cycles and cost.

Performance Efficiency

Performance Efficiency is the ability of your workload to scale to meet the demands placed on it by users in an efficient manner. For more information, see [Design review checklist for Performance Efficiency](#).

- Consider using Batch to perform decoding if the number and size of recorded data files is more than 1,000 files or 300 MB per day.
- Consider performing common calculations and analysis after ingest and storing them in extra tables.
- Use [KQL query best practices](#) to make your query run faster.
- Use a `where` clause to define a time window to reduce the amount of data that's queried. Consider changing the data partition policy for the signals table if your common search criteria aren't time-based, for instance if you filter by recording ID and signal name. When the KQL database expands to contain billions or trillions of records, proper data filtration becomes essential, especially considering the active [partition policy](#).

Warning

Consult with your support team before you alter a data partition policy.

Deploy this scenario

Use the [step-by-step tutorial](#) to deploy this scenario. The guide shows how to deploy a free instance, parse MDF files, ingest data, and perform several basic queries.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal authors:

- [Boris Scholl](#) | Partner, Chief Architect
- [Frank Kaleck](#) | Industry Advisor Automotive

- [Henning Rauch](#) | Principal Program Manager
- [Mario Ortegon-Cabrera](#) | Principal Program Manager

Other contributors:

- [Devang Shah](#) | Principal Program Manager
- [Hans-Peter Bareiner](#) | Cloud Solution Architect
- [Jason Bouska](#) | Senior Software Engineer, Azure Patterns & Practices

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

- [MQTT broker feature in Event Grid](#)
- [Add a KQL database destination to an eventstream](#)
- [Get data from OneLake](#)
- [Materialized views](#)
- [Create a real-time dashboard](#)
- [Create Data Activator alerts from a real-time dashboard](#)
- [Power BI report](#)
- [Visualize data from Azure Data Explorer in Grafana](#)
- [Automotive messaging, data, and analytics reference architecture](#)

Related resources

- [Software-defined vehicle DevOps toolchain](#)
- [Reference architecture for autonomous vehicle operations \(AVOps\)](#)
- [Claim-Check pattern](#)

Azure Load Testing with custom plugins for Event Hubs and IoT Hub

Article • 03/20/2025

💡 Solution ideas

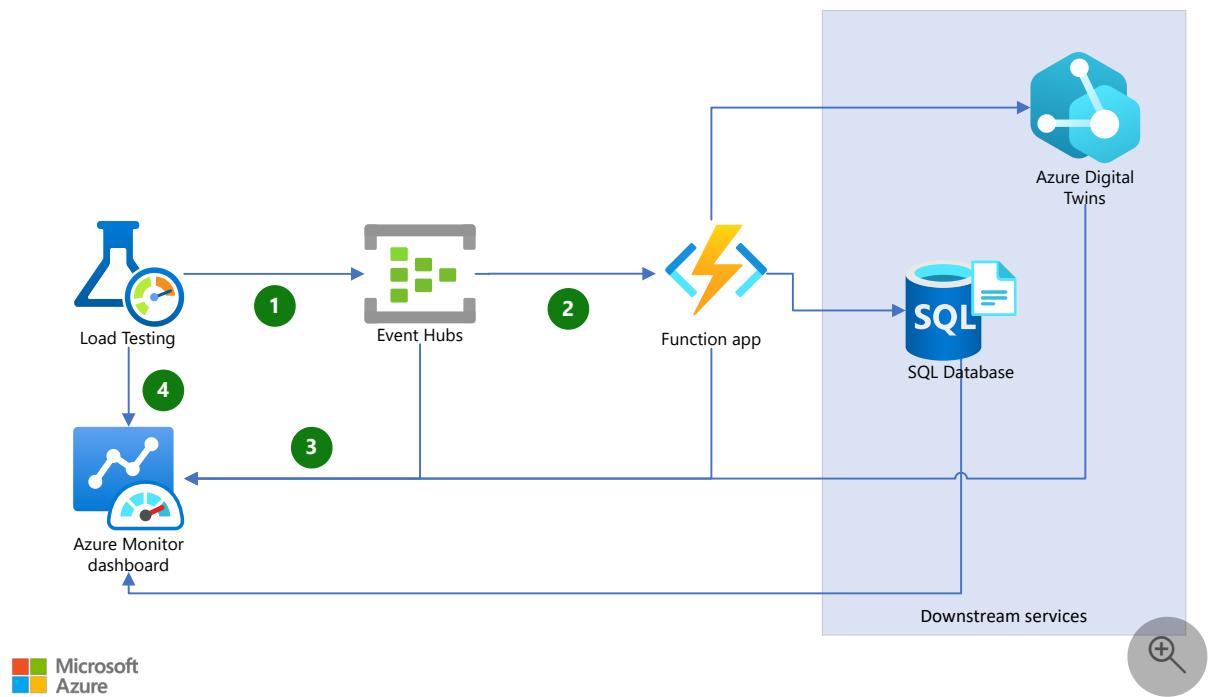
This article describes a solution idea. Your cloud architect can use this guidance to help visualize the major components for a typical implementation of this architecture. Use this article as a starting point to design a well-architected solution that aligns with your workload's specific requirements.

This solution provides guidance for how to use Azure Load Testing, a service that lets you run Apache JMeter scripts and custom plugins to simulate user and device behaviors. This solution also explains how to design key performance indicators (KPIs) and develop a dashboard for monitoring and analyzing the results of the load test in a sample application with Azure Functions and Azure Event Hubs. This example uses Event Hubs, but you can apply the same approach to Azure IoT Hub by changing the Event Hubs client to the IoT Hub client. This article assumes that you have some familiarity with JMeter, its plugins and custom plugins, and Functions and Event Hubs.

IoT Hub contains more core components than Event Hubs, including partitions. Therefore the load testing approach described in this article also applies to IoT Hub with minimal changes.

Architecture

To perform load testing, you need a test plan, which is a set of instructions that tells JMeter what to do during the test. The test plan can include multiple test scenarios, and each scenario can have different settings and configurations. For example, you might have one scenario that simulates a single user accessing a web application and another scenario that simulates multiple users simultaneously accessing the same application.



Download a [Visio file](#) of this architecture.

Dataflow

The following dataflow corresponds to the previous diagram:

1. A simulated device sends data to an event hub through a Load Testing agent. Any behavior of the device can be simulated by using JMeter custom plugins. The Load Testing agent sends data to the event hub after it runs the custom plugin for any type of simulated device.
2. The event hub triggers an Azure function app that processes the data and sends it to other downstream services, such as Azure SQL Database and Azure Digital Twins.
3. Azure Monitor monitors the function app and Event Hubs.
4. Load Testing collects the data from Azure Monitor and displays it in a dashboard.

Components

This example uses the following components:

- **Load Testing:** Use Load Testing to run Apache JMeter scripts and custom plugins to simulate user and device behaviors. Load Testing provides a web-based interface to manage and run load tests and a set of APIs to automate the process. Load Testing is a fully managed service, which means that you don't need to

manage servers or infrastructure. In this architecture, Load Testing uploads the JMeter scripts and custom plugins, and it's the compute where those scripts run.

- **Event Hubs:** Event Hubs is a cloud-based event processing service that you can use to collect, process, and analyze events and streaming data from various sources in real time. Event Hubs supports multiple protocols, including Advanced Message Queuing Protocol (AMQP), HTTPS, Kafka protocol, Message Queuing Telemetry Transport (MQTT), and AMQP over WebSocket. This architecture is event based, so Load Testing populates events to load test the workload.
- **IoT Hub:** IoT Hub is a cloud-hosted managed service that serves as a central message hub for communication between an IoT application and its attached devices. You can connect millions of devices and their back-end solutions reliably and securely. Almost any device can be connected to an IoT hub. You can adapt this architecture to use IoT Hub by changing the Event Hubs client to the IoT Hub client.
- **Azure Functions:** Functions is a serverless compute service that you can use to run code without needing to manage servers or infrastructure. It supports multiple programming languages, including C#, F#, Java, JavaScript, PowerShell, Python, and TypeScript. This architecture uses Functions as the primary compute tier. Event data in Event Hubs triggers and scales out function apps.
- **JMeter GUI**: JMeter GUI is an open-source load testing tool that's primarily used to test the performance of web applications. However, you can also use it to test other types of applications, such as SOAP and REST web services, FTP servers, and databases.
- **Azure Monitor:** Azure Monitor provides monitoring and alerting capabilities for Azure resources. Use it to monitor the performance and health of your applications and the underlying infrastructure. In this architecture, Azure Monitor monitors the health of Azure infrastructure components during the load test.

Scenario details

You can use Load Testing to apply an existing Apache JMeter script and run a load test at cloud scale on any Azure resource.

JMeter lets testers create and run load tests, stress tests, and functional tests. It simulates multiple users simultaneously accessing a web application so that testers can identify potential performance bottlenecks or other problems that might arise under

heavy loads. You can use JMeter to measure various performance metrics, such as response time, throughput, and error rate.

JMeter uses a GUI-based interface to allow users to create test plans, which can include multiple test scenarios that have different settings and configurations. Testers can also customize JMeter by using plugins or by writing custom code. The plugins can help users work with services that use non-HTTP protocols, such as AMQP and WebSocket.

JMeter provides a wide range of features and functions for load testing, but the built-in functionality might not cover specific use cases or requirements. By developing custom plugins, testers can add new functionality or customize existing features to better suit their needs.

For example, you can develop a custom plugin to simulate a specific type of user behavior or to generate more realistic test data. You can also develop custom plugins to integrate JMeter with other tools or systems, such as logging and reporting tools or continuous integration and continuous deployment (CI/CD) pipelines. The custom plugins can help streamline the testing process and make it easier to incorporate load testing into the overall software development workflow. They allow testers to tailor JMeter to their specific needs and improve the accuracy and effectiveness of their load testing efforts.

In this example, a device reports temperature and humidity over a specific period of time. The example simulates this behavior by using a custom JMeter plugin. The [current implementation of the custom plugin](#) generates random data by using a provided template. However, the plugin can contain any possible complex behavior for any device. In this example, the device sends the data to an event hub in Azure. The event hub triggers an Azure function app that processes the data and sends it to other downstream services, such as SQL Database. Azure Functions is the service that the architecture tests. The test plan is designed to simulate the behavior of the device and send data to the event hub.

Potential use cases

Using Load Testing with custom plugins can be useful in various scenarios, such as:

- Testing the performance of an application that uses non-HTTP protocols, such as AMQP and WebSocket.
- Testing the performance of an application that uses a custom protocol.
- Testing the performance of an application that uses a non-Microsoft SDK.
- Simulating a specific type of user or device behavior.
- Generating more realistic test data.

Custom plugins

In JMeter, custom plugins are software components that you can add to expand its default functionality. Custom plugins add new features, functions, or integrations to JMeter. You can develop custom plugins by using the Java programming language and the JMeter plugin development kit (PDK). The PDK provides a set of tools and APIs that help you create new plugins, including GUI elements, listeners, and samplers.

To implement a custom sampler for Event Hubs in JMeter, follow the instructions in [Load Testing JMeter plugins](#). After you implement your custom sampler, you can use it in your JMeter test plan in Load Testing just like any other sampler.

You can implement a test plan by using a thread group that controls the number of threads, like virtual users and devices, to run a specific scenario. Each thread group can have different settings for the number of threads, ramp-up period, loop count, and duration. Thread groups can run sequentially or in parallel, depending on the test plan configuration and the application requirements. You can add the sampler to a thread group, set its parameters, and configure it as needed. Custom samplers can be powerful tools in JMeter that allow you to simulate complex scenarios and requests that the built-in samplers don't support.

Create an Apache JMeter script with a custom plugin

In this section, you create a sample JMeter test script to load test an application with Event Hubs.

To create a sample JMeter test script:

1. Create a *LoadTest.jmx* file in a text editor and paste the following code snippet into the file. This script simulates a load test of 36 virtual machines that simultaneously send events to an event hub. It takes 10 minutes to complete.

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<jmeterTestPlan version="1.2" properties="5.0" jmeter="5.5">
  <hashTree>
    <TestPlan guiclass="TestPlanGui" testclass="TestPlan"
      testname="Test Plan" enabled="true">
      <stringProp name="TestPlan.comments"></stringProp>
      <boolProp name="TestPlan.functional_mode">false</boolProp>
      <boolProp name="TestPlan.tearDown_on_shutdown">true</boolProp>
      <boolProp
        name="TestPlan.serialize_threadgroups">false</boolProp>
        <elementProp name="TestPlan.user_defined_variables"
          elementType="Arguments" guiclass="ArgumentsPanel" testclass="Arguments">
```

```

        testname="User Defined Variables" enabled="true">
            <collectionProp name="Arguments.arguments"/>
        </elementProp>
        <stringProp name="TestPlan.user_define_classpath"></stringProp>
    </TestPlan>
    <hashTree>
        <ThreadGroup guiclass="ThreadGroupGui" testclass="ThreadGroup"
testname="Thread Group" enabled="true">
            <stringProp
name="ThreadGroup.on_sample_error">continue</stringProp>
            <elementProp name="ThreadGroup.main_controller"
elementType="LoopController" guiclass="LoopControlPanel"
testclass="LoopController" testname="Loop Controller" enabled="true">
                <boolProp
name="LoopController.continue_forever">false</boolProp>
                <intProp name="LoopController.loops">-1</intProp>
            </elementProp>
            <stringProp name="ThreadGroup.num_threads">36</stringProp>
            <stringProp name="ThreadGroup.ramp_time">20</stringProp>
            <boolProp name="ThreadGroup.scheduler">true</boolProp>
            <stringProp name="ThreadGroup.duration">600</stringProp>
            <stringProp name="ThreadGroup.delay"></stringProp>
            <boolProp
name="ThreadGroup.same_user_on_next_iteration">false</boolProp>
        </ThreadGroup>
        <hashTree>
            <com.microsoft.eventhubplugin.EventHubPlugin
guiclass="com.microsoft.eventhubplugin.EventHubPluginGui"
estclass="com.microsoft.eventhubplugin.EventHubPlugin" testname="Azure
Event Hubs Sampler" enabled="true">
                <!-- Azure Event Hub connection configuration using
Managed Identity (see repository for instructions) -->
                <boolProp name="useManagedIdentity">true</boolProp>
                <stringProp name="eventHubNamespace">telemetry-
ehn.servicebus.windows.net</stringProp>
                <stringProp name="eventHubName">telemetry-data-changed-
eh</stringProp>
                <stringProp
name="liquidTemplateFileName">StreamingDataTemplate.liquid</stringProp>
            </com.microsoft.eventhubplugin.EventHubPlugin>
        </hashTree>
    </hashTree>
</hashTree>
</jmeterTestPlan>

```

The implementation of `com.microsoft.eventhubplugin.EventHubPluginGui` and `com.microsoft.eventhubplugin.EventHubPlugin` are available at [Azure samples](#).

2. In the file, set the Event Hubs connection values by using the assigned managed identity of the test runners. That identity needs to have write access to the Event Hubs instance.

3. In the file, set the value of the `eventHubName` node to the event hub name, such as `telemetry-data-changed-eh`.

4. Set the value of the `liquidTemplateFileName` node to the file that contains the message that's sent to the event hub. For example, create a file named `StreamingDataTemplate.liquid` as:

JSON

In this example, the payload for the event hub message is a JSON object that has three properties, `MachineId`, `Temperature`, and `Humidity`. `MachineId` is a randomly generated ID that's 27 characters long, and `Temperature` and `Humidity` are random integers that are less than 100. This file is a liquid template syntax. Liquid template is a popular templating language that's used in various web development frameworks. Liquid templates enable developers to create dynamic content that can be easily updated and modified. They allow you to insert variables, conditions, loops, and other dynamic elements into your event hub messages. The syntax is straightforward, and there are plenty of online resources available to help you get started. Liquid templates provide a powerful and flexible way to create dynamic, customizable messages.

5. Save and close the file.

ⓘ Important

Don't include any personal data in the sampler name in the JMeter script. The sampler names appear in the Load Testing test results dashboard. A sample of a liquid template along with the JMeter script file is available to download at [Azure samples](#).

Update the custom plugin from Event Hubs to IoT Hub

The custom plugin uses Event Hubs as the primary target resource. The following configuration is the SDK client setup for Event Hubs:

Java

```
EventHubProducerClient producer = null;
EventHubClientBuilder producerBuilder = new EventHubClientBuilder();

producerBuilder.credential(getEventHubNamespace(), getEventHubName(), new
DefaultAzureCredentialBuilder().build());
producer = producerBuilder.buildProducerClient();

EventDataBatch batch = producer.createBatch(new CreateBatchOptions());
batch.tryAdd(new EventData(msg));
producer.send(batch);
```

The following example shows how you can reuse the same solution, add the IoT dependencies, and change the SDK client to use IoT Hub:

Java

```
IotHubServiceClientProtocol protocol = IotHubServiceClientProtocol.AMQPS;
ServiceClient client = new ServiceClient(getIoTHostName(), new
DefaultAzureCredentialBuilder().build(), protocol);
client.open();

Message message = new Message(msg);
client.send(getDeviceName(), message);

client.close();
```

Run the load test by using the new plugin

When Load Testing starts your load test, it first deploys the JMeter script along with all the other files onto test engine instances. Before running the test, go to the parameter tab to define and set any required parameters. For more information, see [Customize a load test with Apache JMeter plugins and Load Testing](#).

Set up performance testing for the environment

For performance testing, it's important that your test environment is similar to the production environment. This example uses the following environment for performance testing to better understand the system's capacity and performance.

[] Expand table

Service	Configuration
Event Hubs	Premium with one processing unit
Azure Functions	Linux with Premium Plan (EP1) - 210 ACU, 3.5 GB Memory, and 1 vCPU equivalent Standard_D1_v2
Region	East US

Choosing the right service tier for any Azure service, including Event Hubs and Azure Functions, is a complex process and depends on many factors. For more information, see [Event Hubs pricing](#) and [Functions pricing](#).

Design KPIs for performance testing

Before you can design KPIs for performance testing, you need to define the business requirements and the system architecture. The business requirements tell you which KPIs, such as response time, throughput, or error rate, that you want to measure. The system architecture tells you how to test the performance of each component, such as web servers, databases, or APIs. It also helps you choose the best performance testing strategy, such as load testing, stress testing, or endurance testing.

This example has the following business requirements:

- The system can handle 1,000 requests per second.
- The system reliability is higher than 0.99.
- The system can handle 1,000 concurrent devices reporting their personal data information.
- You can specify the maximum capacity of the system in terms of the number of devices that it can support. For example, the system can support 1,000 concurrent devices with three times the current capacity.

To measure whether the system meets these requirements, you can use the following KPIs for performance testing:

[] [Expand table](#)

KPI	Description
RPS	Requests per second for an event hub
LOAD	Number of loads or requests sent to the event hub during performance testing
IR	Number of function invocations or ingestion rate

KPI	Description
RT	Average time for Azure Functions run time
AMU	Average memory usage for Azure Functions
SR	Success rate of all function app invocations
ARS	Average downstream service response time for services like SQL server or a microservice
DF	Dependency failure count, including internal function app errors
MRPS	Maximum RPS with no backlog in the event hub (system capacity)

Measure KPIs

To measure KPIs, you need to have a performance testing strategy. The strategy defines the performance testing approach for each component. This example uses the following performance testing strategy.

- **Event Hubs:** The performance testing approach for the event hub is to send many messages to the event hub and then measure the RPS and LOAD. The RPS is the number of messages that are sent to the event hub per second. The LOAD is the total number of messages that are sent to the event hub during the performance testing. Load Testing can measure RPS and LOAD.
- **Azure Functions:** The performance testing approach for Functions is to measure the following metrics.
 - IR
 - RT
 - AMU
 - SR
 - ARS
 - DF

Azure Monitor can measure AMU, ARS, and DF, but not IR, RT, or SR. To measure KPIs by using Azure Monitor, enable Application Insights for Azure Functions. For more information, see [Enable Application Insights integration](#).

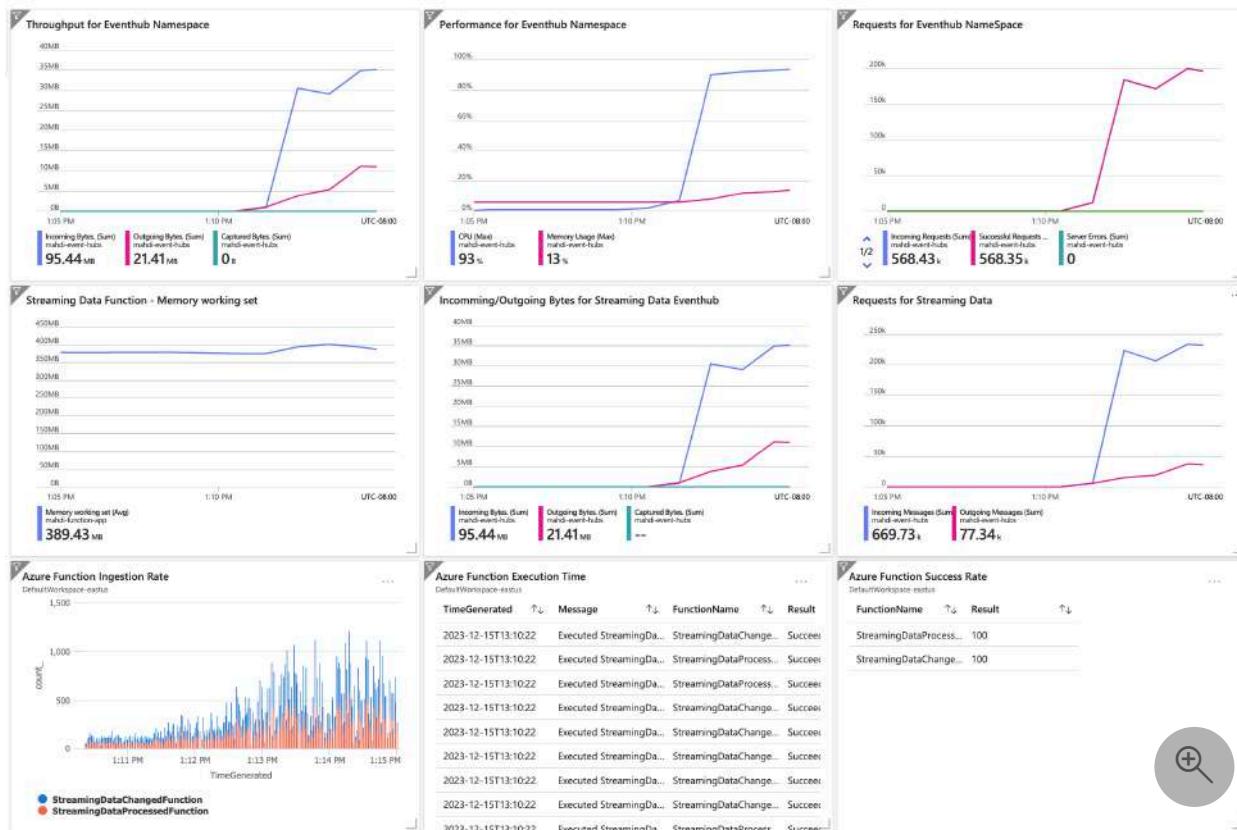
After you enable Azure Monitor, you can use the following queries to measure KPIs:

- IR: `FunctionAppLogs | where Category startswith "name-space-of-your-function" and Message startswith "Executed" | summarize count() by FunctionName, Level, bin(TimeGenerated, 1h) | order by FunctionName desc`

- RT: `FunctionAppLogs | where Category startswith "name-space-of-your-function" and Message startswith "Executed " | parse Message with "Executed " Name "(" Result ", Id=" Id ", Duration=" Duration:long "ms)" | project TimeGenerated, Message, FunctionName, Result, FunctionInvocationId, Duration`
- SR: `FunctionAppLogs | where Category startswith "name-space-of-your-function" and Message startswith "Executed" | summarize Success=countif(Level == "Information"), Total=count() by FunctionName| extend Result=Success*100.0/Total| project FunctionName, Result| order by FunctionName desc`

Azure Monitor dashboard sample

The following image shows a sample of the Azure Monitor dashboard. It shows the KPIs for Azure Functions based on the previous queries.



Conclusion

In this article, you learned how to design KPIs and develop a dashboard for Load Testing. You also learned how to use custom plugins in JMeter to perform load testing on Azure Functions integrated with Event Hubs. You can use the same approach to perform load testing on other Azure services. You can also set up a CI/CD pipeline for your load testing scripts by using Azure DevOps.

For more information, see [Load Testing](#).

Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal authors:

- [Mahdi Setayesh](#) | Principal Software Engineer
- [Oscar Fimbres](#) | Senior Software Engineer

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

- [Load Testing](#)
- [Sample code for a custom JMeter plugin](#)
- [How to develop a new custom plugin](#)
- [Customize a load test with Apache JMeter plugins and Load Testing](#)
- [What is Application Insights](#)
- [Load testing your Azure App Service applications](#)
- [Quickstart: Create and run a load test with Load Testing](#)
- [Load test a website by using a JMeter script in Load Testing](#)
- [Quickstart: Automate an existing load test with CI/CD](#)
- [Tutorial: Run a load test to identify performance bottlenecks in a web app](#)

Feedback

Was this page helpful?

 Yes

 No

IoT analytics with Azure Data Explorer and Azure IoT Hub

Azure Cosmos DB

Azure Data Explorer

Azure Digital Twins

Azure IoT Hub

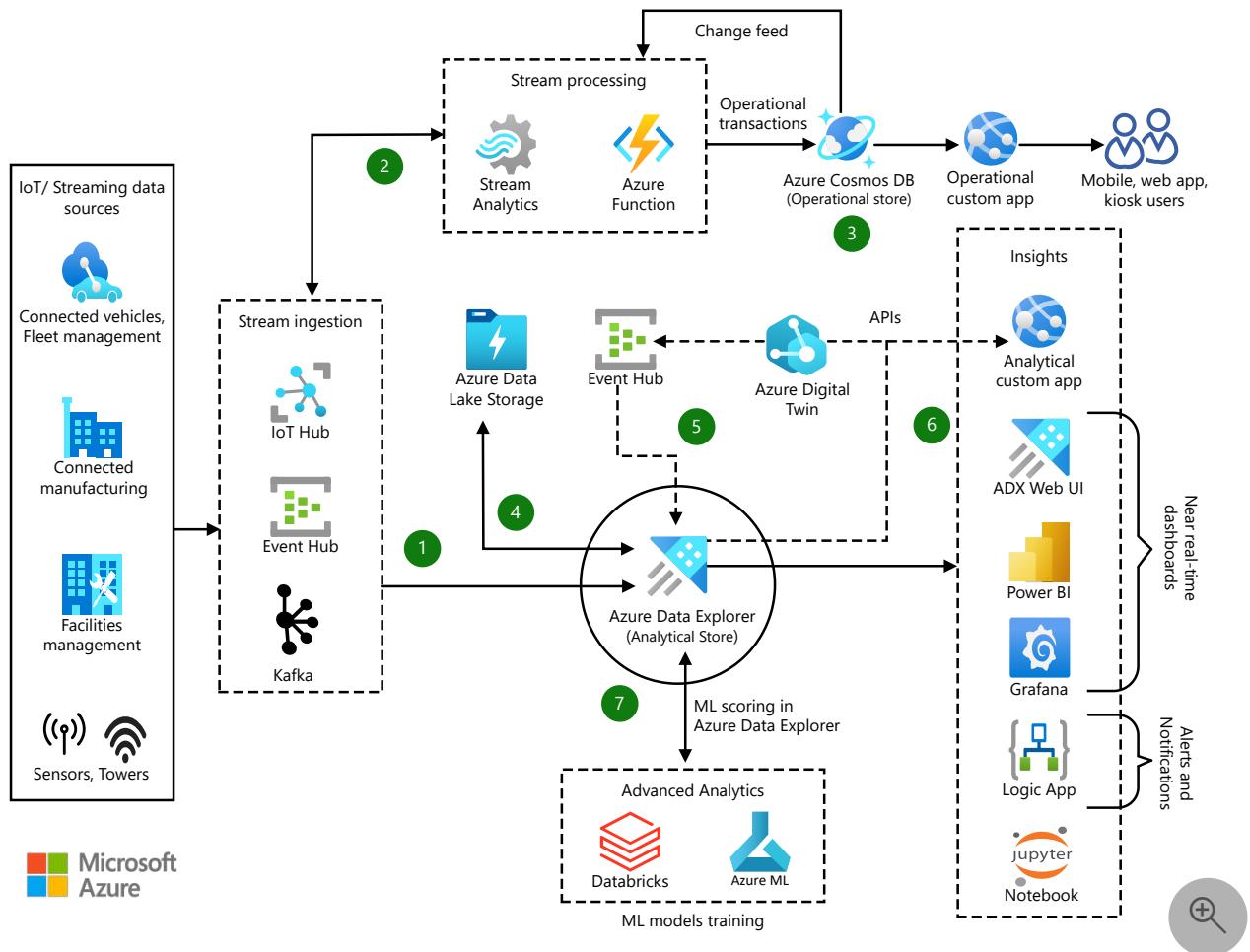
Solution ideas

This article describes a solution idea. Your cloud architect can use this guidance to help visualize the major components for a typical implementation of this architecture. Use this article as a starting point to design a well-architected solution that aligns with your workload's specific requirements.

This solution idea describes how Azure Data Explorer provides near real-time analytics for fast flowing, high volume streaming data from internet of things (IoT) devices and sensors. This analytics workflow is part of an overall IoT solution that integrates operational and analytical workloads with Azure Cosmos DB and Azure Data Explorer.

Jupyter is a trademark of its respective company. No endorsement is implied by the use of this mark. Apache® and Apache Kafka® are either registered trademarks or trademarks of the Apache Software Foundation in the United States and/or other countries. No endorsement by The Apache Software Foundation is implied by the use of these marks.

Architecture



Download a [Visio file](#) of this architecture.

Dataflow

1. Azure Event Hubs, Azure IoT Hub, or Kafka ingest a wide variety of fast-flowing streaming data such as logs, business events, and user activities.
2. Azure Functions or Azure Stream Analytics process the data in near real time.
3. Azure Cosmos DB stores streamed messages in JSON format to serve a real-time operational application.
4. Azure Data Explorer ingests data for analytics, using its connectors for [Azure Event Hubs](#), [Azure IoT Hub](#), or [Kafka](#) for low latency and high throughput.

Alternatively, you can ingest blobs from your [Azure Blob Storage](#) or [Azure Data Lake Storage](#) account into Azure Data Explorer by using an [Event Grid data connection](#).

You can also continuously export data to Azure Storage in compressed, partitioned [Apache Parquet](#) format, and seamlessly query the data with Azure Data Explorer. For more information, see [Continuous data export overview](#).

5. To serve both the operational and analytical use cases, data can either route to Azure Data Explorer and Azure Cosmos DB in parallel, or from Azure Cosmos DB to Azure Data Explorer.

- Azure Cosmos DB transactions can trigger Azure Functions via change feed. Functions will stream data to Event Hubs for ingestion into Azure Data Explorer.

-or-

- Azure Functions can invoke Azure Digital Twins through its API, which then streams data to Event Hubs for ingestion into Azure Data Explorer.

6. The following interfaces get insights from data stored in Azure Data Explorer:

- Custom analytics apps that blend data from Azure Digital Twins and Azure Data Explorer APIs
- Near real-time analytics dashboards that use Azure Data Explorer dashboards, [Power BI](#), or [Grafana](#)
- Alerts and notifications from the [Azure Data Explorer connector for Azure Logic Apps](#)
- The Azure Data Explorer Web UI, [Kusto.Explorer](#), and [Jupyter notebooks](#)

7. Azure Data Explorer integrates with [Azure Databricks](#) and [Azure Machine Learning](#) to provide machine learning (ML) services. You can also build ML models using other tools and services, and export them to Azure Data Explorer for scoring data.

Components

This solution idea uses the following Azure components.

Azure Data Explorer

- [Anomaly detection and forecasting](#) is a built-in analytics feature in [Azure Data Explorer](#). It detects outliers and predicts future values to support proactive monitoring and decision-making. In this architecture, it identifies unusual patterns in IoT telemetry and forecasts expected behavior over time.
- [Anomaly diagnosis for root analysis](#) is a Kusto Query Language (KQL) capability that helps identify the root causes of anomalies. It analyzes contributing dimensions and metrics to streamline troubleshooting. In this architecture, it isolates the source of anomalies detected in device data.

- [Azure Data Explorer](#) is a fully managed, high-performance analytics service. It processes large volumes of streaming data from applications, websites, and IoT devices in near real-time. In this architecture, it serves as the central analytics engine for ingesting, querying, and visualizing IoT data.
- [Azure Data Explorer dashboards](#) are a visualization feature within the Web UI. They allow users to export Kusto queries into interactive dashboards for real-time data exploration. In this architecture, they display insights from IoT data streams and anomaly detection results.
- [Azure Data Explorer web UI](#) is a browser-based interface for working with Azure Data Explorer clusters. It supports writing, running, and sharing KQL commands and queries. In this architecture, it provides a workspace for analysts to query and explore IoT telemetry.
- [Time series analysis](#) is a built-in capability in Azure Data Explorer. It enables users to explore temporal patterns, trends, and seasonality in time-based data. In this architecture, it reveals long-term trends and cyclical behavior in IoT sensor readings.

Other Azure components

- [Azure Cosmos DB](#) is a fully managed, fast NoSQL database service for modern app development with open APIs for any scale. In this architecture, it stores operational data from IoT devices for scalable, low-latency access.
- [Azure Digital Twins](#) is a platform for modeling physical environments as digital representations. In this architecture, it maintains digital models of IoT-connected assets to support spatial analysis and contextual insights.
- [Azure IoT Hub](#) enables bi-directional communication between IoT devices and the Azure cloud. In this architecture, it serves as the central messaging hub for device telemetry and command-and-control operations.
- [Event Hubs](#) is a fully managed, real-time data ingestion service. In this architecture, it ingests telemetry from IoT devices and streams it into the analytics pipeline.
- [Kafka on HDInsight](#) is an enterprise-grade, cost-effective service for running Apache Kafka on Azure. In this architecture, it provides an alternative streaming backbone for ingesting and distributing IoT data.

Scenario details

This solution uses Azure Data Explorer to get near real-time IoT telemetry analytics on fast-flowing, high-volume streaming data from a wide variety of IoT devices.

Potential use cases

- Fleet management, for predictive maintenance of vehicle parts. This solution is ideal for the automotive and transportation industry.
- Facilities management, for energy and environment optimization.
- Combining real-time road conditions with weather data for safer autonomous driving.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Shlomo Sagir](#) | Senior Content Developer

Next steps

- [What is Azure Data Explorer?](#)
- [Visualize data with Azure Data Explorer dashboards](#)

Related resource

- [Analytics architecture design](#)

Use Azure IoT Hub to privately upload files to an Azure Storage account

Azure IoT Hub

Azure Storage

Azure Firewall

Azure Virtual Network

Azure Application Gateway

Solution ideas

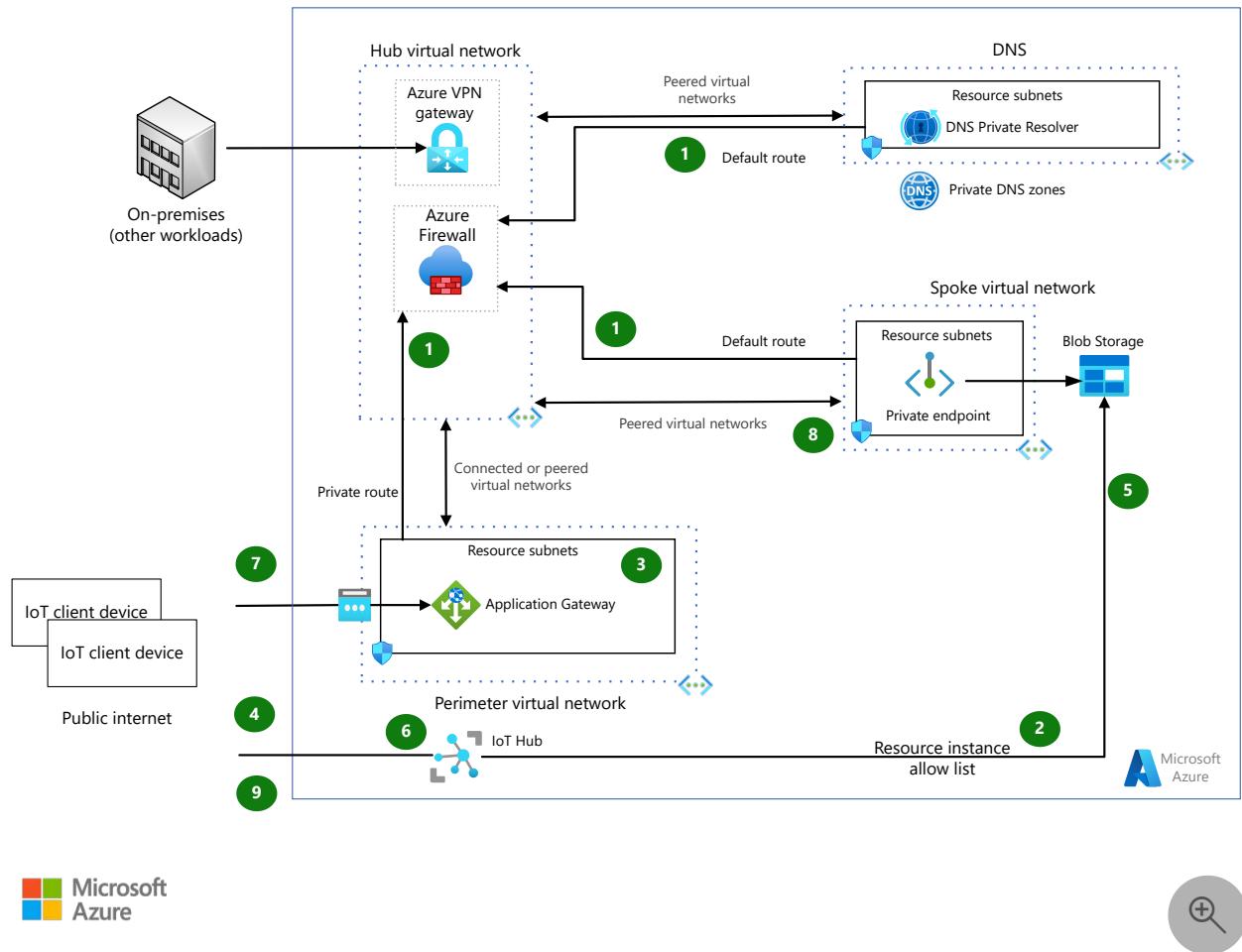
This article describes a solution idea. Your cloud architect can use this guidance to help visualize the major components for a typical implementation of this architecture. Use this article as a starting point to design a well-architected solution that aligns with your workload's specific requirements.

This article describes how to use a private network to upload files to an Azure Storage account.

For typical Azure IoT deployments, the IoT client devices need to communicate directly with the Storage account to upload files. IoT client devices are typically distributed at disparate locations, and they aren't part of a private network, so they connect over the public internet. You can't integrate these devices into a private network, so the Storage account requires that you allow incoming internet traffic.

But if you have stricter network segmentation requirements, you can restrict access to the Storage account from within a private network. This solution blocks direct internet traffic to the Storage account so that the Storage account only accepts traffic that goes through the inbound Azure Application Gateway instance. If you implement a [hub-spoke network topology](#), Azure Firewall typically must inspect traffic, which provides an extra layer of security.

Architecture



Download a [Visio file](#) of this architecture.

Workflow

The following workflow corresponds to the preceding diagram.

1. A hub-spoke network topology has a hub virtual network that peers to each resource virtual network, also called a *spoke*. All traffic goes through Azure Firewall for traffic inspection.
2. An Azure Blob Storage account denies public internet access. It only allows connections from other virtual networks. A [resource instance rule](#) allows a chosen Azure IoT Hub service to connect via a managed identity. The Blob Storage account only supports Azure role-based access control (RBAC) for data plane access.
3. The application gateway has custom Domain Name System (DNS) and terminates Transport Layer Security (TLS) traffic. It resides within a virtual network. This virtual network peers with the virtual network that the Blob Storage account's private link uses. A forced tunnel via the hub virtual network establishes the connection.

4. The IoT client device that uses the IoT Hub SDK requests a shared access signature (SAS) URI for file uploads to IoT Hub. The IoT client device sends the request through the public internet.

5. IoT Hub handles this request for the device. It connects directly to the Blob Storage account via managed identity authentication, which has *Storage Blob Data Contributor* permissions for user-delegation key requests.

IoT Hub requests a user-delegation key to the Blob Storage account. A short-lived SAS token grants the device read-write permission on the requested blob in the blob container.

6. IoT Hub sends the public Blob Storage account URI and SAS token to the IoT client device, along with a correlation ID.

7. The IoT client device has logic to replace the public Blob Storage URI with a custom domain, for example a [device twin](#). The IoT device uses a standard Blob Storage SDK to upload the file through the custom Blob Storage DNS.

8. Application Gateway receives the HTTP POST from the client device and sends it to the Blob Storage account via Azure Private Link.

9. When the file upload is finished, the IoT client device uses the Azure IoT SDK to notify IoT Hub.

The IoT client device updates the file upload status so that IoT Hub can trigger a file upload notification to back-end services, if the notification is configured. The client device also releases resources that are associated with the file upload in IoT Hub.

Components

- [Application Gateway](#) is a platform as a service (PaaS)-managed solution that you can use to build highly secure, scalable, and highly available front ends. In this architecture, Application Gateway handles the incoming internet HTTPS traffic, applies TLS termination, negotiates TLS with the Blob Storage account, and forwards traffic through a private network to the Blob Storage account.
- [Azure Firewall](#) provides protection for your Azure Virtual Network resources. In this architecture, Azure Firewall filters and routes traffic between the perimeter network and spoke networks.
- [IoT Hub](#) is a PaaS-managed solution that acts as a central message hub for bidirectional communication between an IoT application and the devices that it manages.

In this architecture, IoT Hub is the central endpoint that IoT client devices connect to for control and data plane operations.

- [Private Link](#) provides private access to services that are hosted on the Azure platform while keeping your data on the Microsoft network. In this architecture, Private Link provides private communication between Application Gateway and the Blob Storage account.
- [Storage](#) offers a durable, highly available, and massively scalable cloud storage solution. It includes object, file, disk, queue, and table storage capabilities. In this architecture, devices use Blob Storage to upload files to the cloud via short-lived SAS tokens that IoT Hub provides through user delegation.
- [Private DNS zones](#) provide a reliable, enhanced-security DNS service to manage and resolve domain names in a virtual network without the need for a custom DNS solution. In this architecture, a private DNS zone provides a private DNS entry for Blob Storage so that the Storage blob endpoint translates to its private IP endpoint within the network.
- [Virtual Network](#) is the fundamental building block for your private network in Azure. This service enables many types of Azure resources, such as Azure virtual machines, to communicate with each other, the internet, and on-premises networks with enhanced security. This architecture uses Virtual Network to build a private network topology, which avoids internet public endpoints for Azure-based services.

Scenario details

For regular deployments, an Azure IoT client device needs to communicate directly to a Storage account to upload a file. Disabling internet traffic on the Storage account blocks any client IoT client devices from uploading files. The IoT Hub file upload functionality acts only as a user delegation for generating a SAS token that has read-write permissions on a blob. The file upload itself doesn't pass through IoT Hub. An IoT client device uses the normal Blob Storage SDK for the actual upload.

In this scenario, communication between IoT Hub and the Storage account still goes through the public endpoint. This exception is possible through Storage networking configurations for resource instances. You can disable public internet access to the Storage account and allow Azure services and specific instances of resources to connect through the Azure backbone. This network perimeter is paired with a Microsoft Entra ID-based identity perimeter that uses Azure RBAC to restrict data plane access.

This architecture assigns a managed identity to IoT Hub. The managed identity is assigned the role of *Storage Blob Data Contributor* to the specified Storage account. With this permission,

IoT Hub can request a user-delegation key to construct a short-lived SAS token. The IoT client device receives the SAS token for the file-upload process.

Application Gateway acts as the entry point for requests that go to the private endpoint of the Storage account, which is configured as the only back end. Application Gateway uses a public IP address. A custom DNS provider can be configured to map the public IP address to an *A* record or *CNAME* record.

If you have internal security requirements to use private endpoints for many Azure PaaS services, you can implement this scenario to provide shorter validation cycles to deploy your IoT solutions in production.

Potential use cases

This architecture can apply to any scenario that uses devices that need to communicate with a Storage account that isn't exposed publicly.

For example, an industrial automation vendor provides managed connected edge controllers and sensors. These sensors need to communicate with the Azure cloud through the public internet, but the vendor's security team requires the Storage account to be denied public internet access. This architecture meets this requirement.

Alternatives

If you don't require the hub-spoke network topology that has Azure Firewall traffic inspection, you can implement a simplified networking topology to benefit from this approach. You could use a single virtual network that has distinct subnets to accommodate Application Gateway, Private Link, and the private DNS zone. The Storage account and IoT Hub can use the same configurations as the original architecture.

The benefits of a simplified architecture include reduced complexity and cost. If you don't have specific business or enterprise requirements for a hub-spoke topology, use the simplified architecture to eliminate public internet endpoints from the Storage account. This approach also helps ensure that IoT applications that use the IoT Hub file upload functionality work correctly.

For a sample that deploys a similar architecture, see [Set up IoT Hub file upload to Storage through a private endpoint](#). This sample deploys a simulated IoT client device and uses device twins to replace the custom domain name for the Storage account.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal authors:

- [Katrien De Graeve](#) | Software Engineer
- [Vincent Misson](#) | Cloud Solution Architect

Other contributor:

- [Nacim Allouache](#) | Cloud Solution Architect

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next step

Learn how to [upload files with IoT Hub](#).

Related resources

- [Hub-spoke network topology in Azure](#)

Project 15 Open Platform IoT sustainability

Azure Event Grid

Azure Event Hubs

Azure Functions

Azure IoT Hub

Azure Stream Analytics

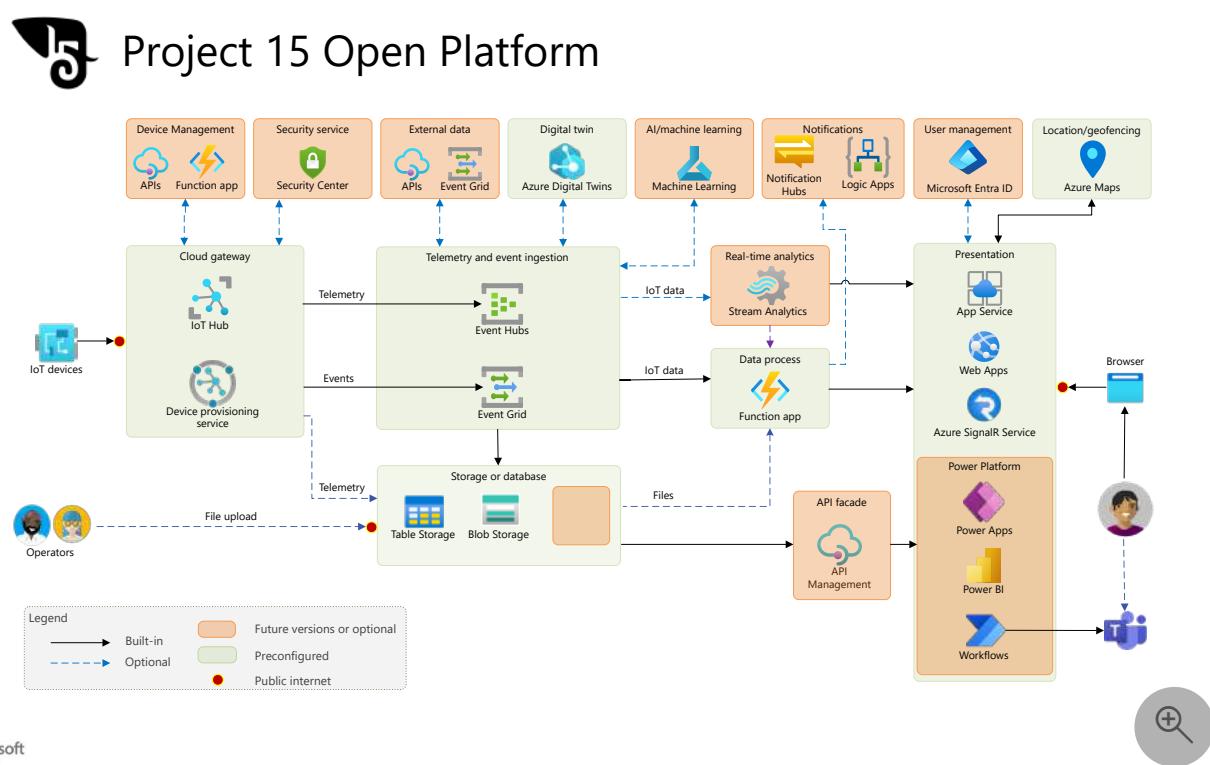
Solution ideas

This article describes a solution idea. Your cloud architect can use this guidance to help visualize the major components for a typical implementation of this architecture. Use this article as a starting point to design a well-architected solution that aligns with your workload's specific requirements.

Project 15 Open Platform was developed in partnership with the GEF Small Grants Programme, which the United Nations Development Programme implemented. For more information, see [Project 15 from Microsoft – A story in five parts](#).

Architecture

The following sections describe the functionality and architecture of Project 15 Open Platform.



Download a [Visio file](#) of this architecture.

Workflow

The following Azure services and configurations make up Project 15 Open Platform:

1. The Azure IoT Hub device provisioning service provisions Internet of Things (IoT) devices and connects them to IoT Hub.
2. Streaming platforms and services build the data pipeline that's necessary for basic telemetry and event processing:
 - Azure Event Hubs ingests telemetry and events from IoT devices.
 - Azure Event Grid provides a publish-subscribe model that routes events.
3. Azure Stream Analytics analyzes data. Azure Functions processes data. Azure Time Series Insights monitors, analyzes, and stores data. These three services also feed data into a presentation layer.
4. Users connect to the presentation layer through browsers. In that layer:
 - Azure SignalR Service messaging provides real-time visualization.
 - Azure App Service and its Web Apps feature provide platforms that you can use to build, deploy, and scale web apps.
 - Tools like Power BI visualize IoT devices, telemetry, and events in websites.
 - Tools like Power Apps and Power Automate provide low-code apps and automated workflows.
5. Databases, Azure Blob Storage, and tables store telemetry and file data from offices in the field.
6. Other Azure components provide more functionality:
 - Azure Functions and Azure API Management work to make device management events available in websites.
 - Microsoft Entra ID manages users.
 - API Management and Event Grid manage external data.
 - Azure Digital Twins provides modeling capabilities that you can use to optimize operations.
 - Microsoft Defender for Cloud secures the solution by establishing security policies and access controls.
 - Azure Notification Hubs and Azure Logic Apps handle notifications.
 - Azure Machine Learning provides AI capabilities to help you predict device behavior.
 - Azure Maps tracks geofencing data to provide location-based services.

Components

- [IoT Hub](#) is a managed service for secure and scalable device-to-cloud communication. In this architecture, it connects devices to Azure cloud resources and enables zero-touch provisioning through its [device provisioning service](#). You can use IoT Hub queries to filter data that you send to the cloud.
- [Event Hubs](#) is a managed big data streaming platform. In this architecture, Event Hubs ingests telemetry and event data from IoT devices to support real-time analytics and processing.
- [Event Grid](#) is a service that enables event-based architectures. In this architecture, it routes events from sources to destinations and decouples event publishers from event subscribers.
- [Stream Analytics](#) is a real-time serverless stream processing service that can run queries in the cloud and on devices on the edge of the network. In this architecture, it filters and aggregates telemetry data from IoT devices for further processing or storage.
- [Functions](#) is an event-driven serverless compute platform that you can use to build and debug locally without extra setup. In this architecture, Functions processes incoming data and integrates services by using triggers and bindings.
- [Azure SignalR Service](#) is an open-source software library that provides a way to send notifications to web apps in real time. In this architecture, SignalR enables live data visualization in the presentation layer.
- [App Service](#) and its [Web Apps](#) feature are managed platforms for building, deploying, and scaling web apps. In this architecture, they support the deployment and scaling of web interfaces for interacting with IoT data.
- [Power BI](#) is a collection of software services and apps that you use to connect and visualize unrelated sources of data. In this architecture, Power BI visualizes telemetry and event data from IoT devices in dashboards and reports.
- [Blob Storage](#) is a service that provides optimized cloud object storage for unstructured data. In this architecture, Blob Storage stores telemetry and file data collected from field devices.
- The [API Apps](#) feature of App Service enables you to use to build and consume APIs in the cloud by using the language of your choice. In this architecture, API Apps exposes device management events and data to external systems.
- [Azure Digital Twins](#) is a modeling service for digital representations of physical environments. In this architecture, Azure Digital Twins models IoT devices and ecosystems to optimize operations, minimize costs, and improve insights.

- [Defender for Cloud](#) is a security management service for hybrid cloud environments. In this architecture, it enforces security policies and protects workloads from threats.
- [Notification Hubs](#) is a push notification service that provides a push engine that you can use to send notifications to any platform from any back end. In this architecture, it sends alerts and updates to users across platforms.
- [Logic Apps](#) is a service that automates workflows and integrates systems, services, and data across organizations by using a visual designer and prebuilt connectors. In this architecture, it orchestrates data flows and integrates services without requiring custom code.
- [Machine Learning](#) is a cloud-based environment that you can use to train, deploy, automate, manage, and track machine learning models. You can use these models to forecast future behavior, outcomes, and trends. In this architecture, Machine Learning predicts device behavior and trends to support proactive decision-making.
- [Azure Maps](#) is a service that provides geospatial capabilities, including mapping, routing, traffic, and location data, to power applications with real-time geographic context. In this architecture, Azure Maps tracks geofencing data and supports location-based insights.
- [Microsoft Power Platform](#) is a low-code development suite for analyzing data, automating processes, and building apps, websites, and virtual agents. In this architecture, it enables rapid development of custom apps and workflows for interacting with IoT data.

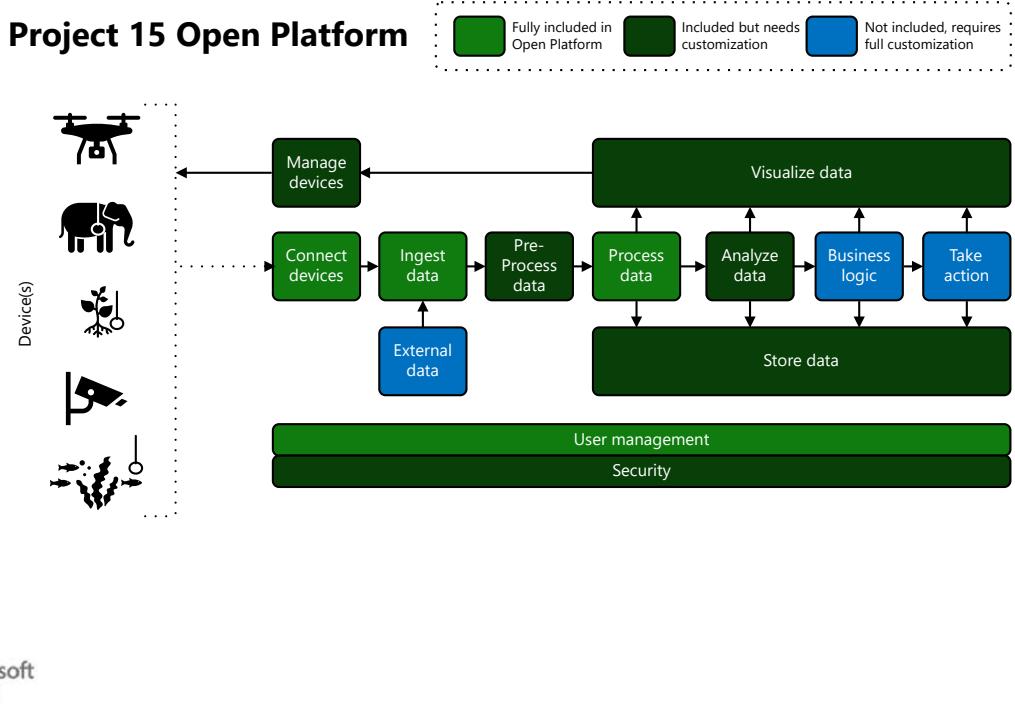
Scenario details

The goal of [Project 15 Open Platform](#) is to bring the latest Microsoft cloud and IoT technologies together to help scientific teams build sustainability and conservation solutions like species tracking and observation, poaching prevention, ecosystem monitoring, and pollution detection.

The core goals of Project 15 Open Platform are to:

- **Close the skills gap, boost innovation, and accelerate problem-solving.** Project 15 Open Platform is a ready-made platform that scientific developers can use for specific scenarios.
- **Decrease the time to deployment.** Project 15 Open Platform gets teams to 80% completion of their projects. This boost dramatically reduces the time that teams need to start making crucial insights.
- **Reduce development costs.** Project 15 Open Platform reduces overall development costs and makes building connected device-based solutions on Azure less complex. The open

platform also gives teams opportunities to partner with the open-source developer community and universities.



Download a [Visio file](#) of this architecture.

Developers at Microsoft currently maintain Project 15 Open Platform, but it isn't an official Microsoft product.

The solution has three main categories:

- **Components that are fully included**

Azure services make up the core infrastructure of the solution. You deploy these services only once, and then you expand them as you add devices to the solution. You don't need to fully understand these services to take advantage of the platform. To get a better understanding of these core components, see the following resources:

- [Internet of Things event learning path](#)
- [Introduction to Azure IoT](#)

- **Components that are included but need customization**

The platform deploys these services for you, but you need to modify them to meet your solution's requirements. For more information on these services, see [Project 15 Open Platform developer guide](#).

- **Components that aren't included and require full customization**

You deploy the services to your own Azure account where you can then customize them to create your solution. Your IP address resides in this account.

Potential use cases

Project 15 Open Platform contributes the latest Azure and IoT technologies to conservation and ecosystem sustainability efforts. These technologies help accelerate scientific innovation in areas like:

- Species tracking and observation
- Poaching prevention
- Ecosystem monitoring
- Pollution detection

Deploy this scenario

Deploy to Azure with the push of a button. The main components of the infrastructure for a standard IoT solution are then up and running.

For more information, see [Deploying Project 15 from Microsoft Open Platform](#).

Contributors

This article is maintained by Microsoft. It was originally written and updated by the following contributors.

Principal authors:

- [Sarah Maston](#) | Director, Global Partner Strategy
- [Daisuke Nakahara](#) | Director, Sony Semiconductor Solutions
- [Linda Nichols](#) | App Innovation Global Black Belt

Next steps

- For more information about deploying to Azure and customizing conservation and ecological sustainability solutions, see [Project 15 Open Platform on GitHub](#).
- [Introduction to Azure IoT](#)
- [Internet of Things event learning path](#)
- [Microsoft and sustainability](#)
- [Seeed Studio's IoT into the wild](#)

Related resources

- IoT architectures

Management and governance architecture design

Article • 02/11/2025

Management and governance includes critical tasks like:

- The monitoring, auditing, and reporting of security and business requirements.
- Implementing backup, disaster recovery, and high availability.
- Ensuring compliance with internal requirements and external regulations.
- The protection of sensitive data.

Azure provides a wide range of services to help you with management and governance.

Here are a few examples:

- [Azure Attestation](#). Remotely verify the trustworthiness of a platform and the integrity of the binaries running inside it.
- [Azure confidential ledger](#). Store and process confidential data with confidence.
- [Azure Purview](#). Govern, protect, and manage your data.
- [Azure Policy](#). Achieve real-time cloud compliance at scale with consistent resource governance.
- [Azure Stack](#). Place technologies and services in appropriate locations, based on your business requirements. Meet custom compliance, sovereignty, and data gravity requirements.
- [Azure Backup](#). Define backup policies and provide protection for a wide range of enterprise workloads.
- [Azure Site Recovery](#). Keep your business running with built-in disaster recovery.
- [Azure Archive Storage](#). Store rarely accessed data.
- [Azure Monitor](#). Get full observability into your applications, infrastructure, and network.
- [Azure Update Manager](#). Centrally manage updates and compliance at scale.

Introduction to management and governance on Azure

If you're new to management and governance on Azure, the best way to learn more is with [Microsoft Learn training](#), a free online training platform. Microsoft Learn provides interactive training for Microsoft products and more.

Here are some resources to get you started:

- Learning path: [Manage information protection and governance](#)
- Module: [Design an enterprise governance strategy](#)
- Module: [Design a solution for backup and disaster recovery](#)

Path to production

The following sections provide links to reference architectures in some key management and governance categories:

Backup

- [Azure Backup architecture and components](#)
- [Support matrix for Azure Backup](#)
- [Backup cloud and on-premises workloads to cloud](#)

Disaster recovery

- [Azure to Azure disaster recovery architecture](#)
- [Support matrix for Azure VM disaster recovery between Azure regions](#)
- [Integrate Azure ExpressRoute with disaster recovery for Azure VMs](#)
- [Move Azure VMs to another Azure region](#)
- [Business continuity and disaster recovery \(BCDR\) for Azure VMware Solution enterprise-scale scenario](#)
- [Use Azure Local stretched clusters for disaster recovery](#)

High availability

- [High availability enterprise deployment using App Service Environment](#)
- [Baseline zone-redundant web application](#)
- [Deploy highly available NVAs](#)
- [Highly available SharePoint farm](#)
- [Recommendations for using availability zones and regions](#)

Compliance and governance

- [Manage virtual machine compliance](#)
- [Introduction of an AKS regulated cluster for PCI-DSS 3.2.1](#)

Hybrid management

- Azure Arc hybrid management and deployment for Kubernetes clusters
- Back up files and applications on Azure Stack Hub
- Enable virtual machine protection in Azure Site Recovery
- Hybrid availability and performance monitoring

Update management

- Plan deployment for updating Windows VMs in Azure

Best practices

The Azure Well-Architected Framework is a set of guiding tenets that you can use to improve the quality of your architectures. For management and governance best practices, see:

- Regulatory compliance
- Administrative account security

For additional guidance, see:

- Design area: [Management for Azure environments](#)
- [Governance best practices](#)

Stay current with management and governance

Get the latest updates on [Azure management](#) and [Azure governance](#) technologies.

Additional resources

Following are a few more management and governance architectures to consider:

- [Management and monitoring for an Azure VMware Solution enterprise-scale scenario](#)
- [Computer forensics chain of custody in Azure](#)

AWS or Google Cloud professionals

- [AWS to Azure services comparison - Management and governance](#)
- [Google Cloud to Azure services comparison - Management](#)

Feedback

Was this page helpful?

 Yes

 No

Plan deployment for updating Windows VMs in Azure

Azure Firewall

Azure Virtual Machines

Azure Virtual Network

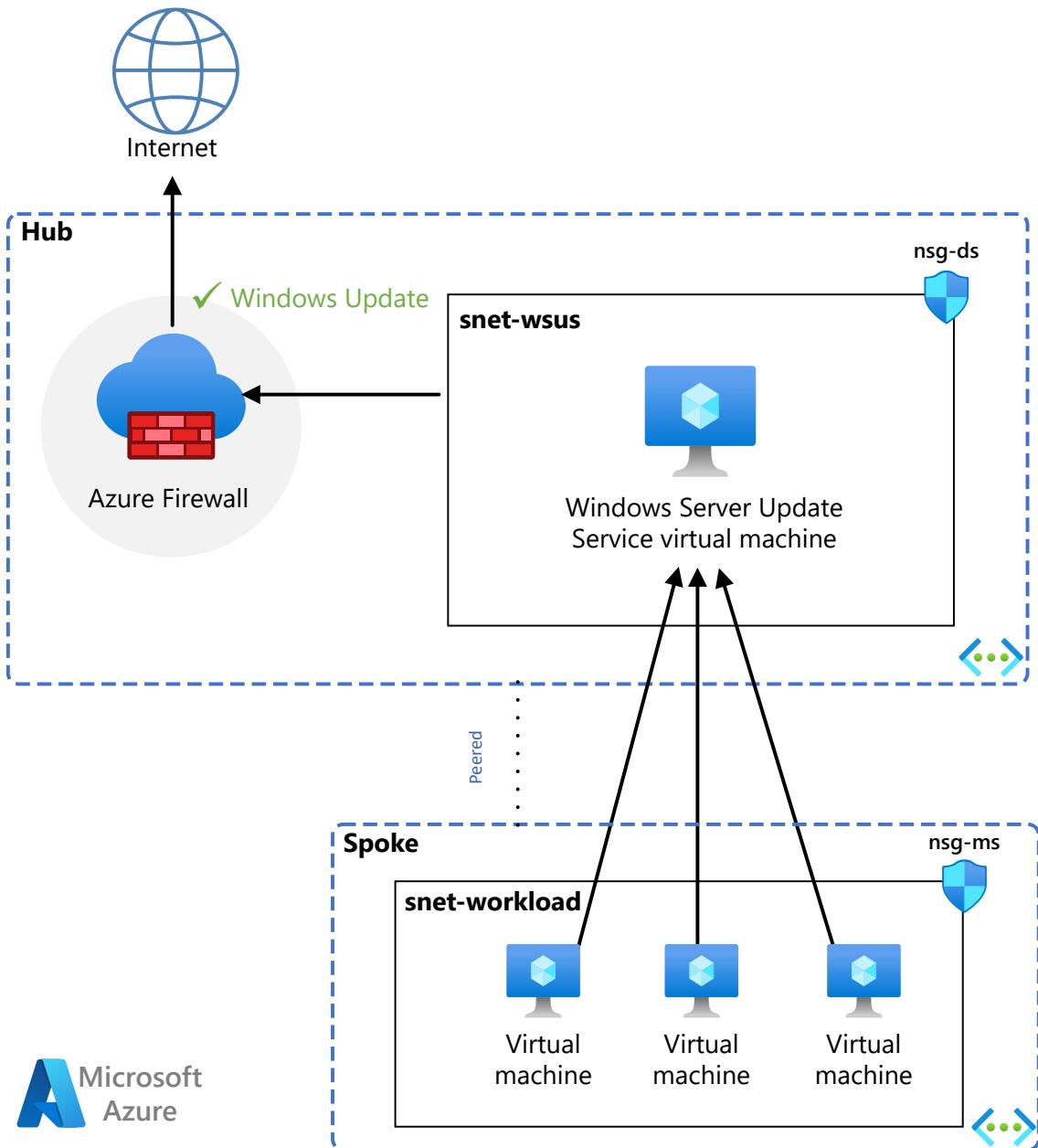
If you've locked down your Azure virtual network from the internet, you can still get Windows updates without jeopardizing security and opening up access to the internet as a whole. This article contains recommendations on how you can set up a perimeter network, also called a DMZ, to host a Windows Server Update Service (WSUS) instance to securely update virtual networks without internet connectivity.

If you're using Azure Firewall, you use the `WindowsUpdate` FQDN tag in application rules to allow the required outbound network traffic through your firewall. For more information, see [FQDN tags overview](#) and [Plan for software updates - Configure firewalls](#).

To implement the recommendations in this article, you should be familiar with Azure services. The following sections describe the recommended deployment design, which uses a hub-spoke configuration in a single region or multiregion configuration.

Azure Virtual Network hub-spoke network topology

We recommend that you set up a hub-spoke model network topology by creating a perimeter network. Host the WSUS server on an Azure virtual machine that's in the hub between the internet and the virtual networks. The hub should have open ports. WSUS uses port 80 for HTTP protocol and port 443 for HTTPS protocol to obtain updates from Microsoft. The spokes are all the other virtual networks, which will communicate with the hub and not with the internet. You can accomplish this by creating a subnet, network security groups (NSGs), and Azure virtual network peering that allows WSUS traffic while blocking other internet traffic. This image illustrates an example of hub-spoke topology:



Download a [Visio file](#) of this architecture.

In this image:

- **snet-wsus** is the subnet in the hub of the hub and spoke topology that contains the WSUS server.
- **nsg-ds** is a network security group rule that allows traffic for WSUS while blocking other internet traffic.
- **Windows Server Update Service virtual machine** is the Azure virtual machine that's configured to run WSUS.
- **snet-workload** is an example of a subnet in a peered spoke virtual network containing Windows virtual machines.

- `nsg-ms` is a network security group policy that allows traffic to the WSUS VM but denies other internet traffic.

You can reuse an existing server or deploy a new one that will be the WSUS server. Your WSUS VM must meet the documented [system requirements](#). As this is a security sensitive capability, you should plan on accessing this virtual machine by using just-in-time (JIT). See [Manage virtual machine access using just-in-time](#).

Your network will have more than one Azure virtual network, which can be in the same region or in different regions. You'll need to evaluate all Windows Server VMs to see if one can be used as a WSUS server. If you have thousands of VMs to update, we recommend dedicating a Windows Server VM to the WSUS role. We also encourage that VMs don't use a WSUS server in a different region as their primary source.

If all your virtual networks are in the same region, we suggest having one WSUS for every 18,000 VMs. This suggestion is based on a combination of the VM requirements, the number of client VMs being updated, and the cost of communicating between virtual networks. For more information on WSUS capacity requirements, see [Plan your WSUS deployment](#).

You can determine the cost of these configurations by using the [Azure pricing calculator](#). You'll need to provide the specifications of your WSUS virtual machines and your network expectations; same region, across regions. For data transfer, start with 3 GB. Note that prices will vary by region.

Manual deployment

After you either identify the Azure virtual network to use or determine you need to create a new Windows Server instance, you need to create an NSG rule. The rule will allow internet traffic, which allows Windows Update metadata and content to sync with the WSUS server that you'll create. Here are the rules that you need to add:

- Inbound/outbound NSG rule to allow traffic to and from the internet on port 80 (for content).
- Inbound/outbound NSG rule to allow traffic to and from the internet on port 443 (for metadata).
- Inbound/outbound NSG rule to allow traffic from the client VMs on port 8530 (default unless configured).

Set up WSUS

There are two approaches you can use to set up your WSUS server:

- If you want to automatically set up a server that's configured to handle a typical workload with minimal administration required, you can use the PowerShell automation script.
- If you need to handle thousands of clients that run many different operating systems and languages, or if you want to configure WSUS in a way that the PowerShell script can't handle, you can set up WSUS manually. Both approaches are described later in this article.

You can also combine the two approaches by using the automation script to do most of the work and then using the WSUS administrative console to fine-tune the server settings.

Set up WSUS by using an automation script

The `Configure-WSUSServer` script allows you to set up a WSUS server that will automatically synchronize and approve updates for a chosen set of products and languages.

ⓘ Note

The script always sets up WSUS to use Windows Internal Database to store its update data. This speeds up setup and reduces administration complexity. But if your server will support thousands of client computers, especially if you also need to support a wide variety of products and languages, you should set up WSUS manually instead so that you can use SQL Server as the database.

The latest version of this script is [available on GitHub](#).

You configure the script by using a JSON file. You can currently configure these options:

- Whether update payloads should be stored locally (and, if so, where they should be stored), or left on the Microsoft servers.
- Which products, update classifications, and languages should be available on the server.
- Whether the server should automatically approve updates for installation or leave updates unapproved unless an administrator approves them.
- Whether the server should automatically retrieve new updates from Microsoft, and, if so, how often.
- Whether Express update packages should be used. (Express update packages reduce server-to-client bandwidth at the expense of client CPU/disk usage and server-to-server bandwidth.)
- Whether the script should overwrite its previous settings. (Normally, to avoid inadvertent reconfiguration that might disrupt server operation, the script will run only once on a given server.)

Copy the script and its configuration file to local storage, and edit the configuration file to suit your needs.

⚠ Warning

Be careful when you edit the configuration file. The syntax used for JSON configuration files is strict. If you inadvertently change the structure of the file rather than just the parameter values, the configuration file won't load.

You can run this script in one of two ways:

- You can run the script manually, from the WSUS VM.

The following command, run from an elevated Command Prompt window, will install and configure WSUS. It will use the script and configuration file in the current directory.

```
powershell.exe -ExecutionPolicy Unrestricted -File .\Configure-WSUSServer.ps1 -  
WSUSConfigJson .\WSUS-Config.json
```

- You can use the [Custom Script Extension for Windows](#).

Copy the script and the JSON configuration file to your own storage container that has private network line of sight to the WSUS VM.

In typical VM and Azure Virtual Network configurations, the Custom Script Extension needs only the following two parameters to run the script correctly. (You need to replace the values shown here with the URLs for your storage locations.)

Bicep

```
settings: {  
  fileUris: [  
    'https://yourstorageaccount.blob.core.windows.net/wsus/Configure-  
    WSUSServer.ps1'  
    'https://yourstorageaccount.blob.core.windows.net/container/WSUS-  
    Config.json'  
  ]  
  commandToExecute: 'powershell.exe -ExecutionPolicy Unrestricted -File  
  .\Configure-WSUSServer.ps1 -WSUSConfigJson .\WSUS-Config.json'  
}
```

The script will start the initial synchronization needed to make updates available to client computers. But it won't wait for that synchronization to complete. Depending on the products, classifications, and languages you've selected, the initial synchronization might take several hours. All synchronizations after that should be faster.

Set up WSUS manually

From your WSUS VM, follow the instructions found in [Install the WSUS Server Role](#)

During synchronization, WSUS determines if any new updates have been made available since the last time you synchronized. If it's your first time synchronizing WSUS, the metadata is downloaded immediately. The payload downloads only if local storage is turned on and the update is approved for at least one computer group.

 **Note**

Initial synchronization can take more than an hour. All synchronizations after that should be significantly faster.

Configure virtual networks to communicate with WSUS

Next, set up Azure virtual network peering or global virtual network peering to communicate with the hub. We recommend that you set up a WSUS server in each region you've deployed to minimize latency.

On each Azure virtual network that's a spoke, you'll need to create an NSG policy that has these rules:

- An inbound/outbound NSG rule to allow traffic to the WSUS VM on port 8530 (default unless configured).
- An inbound/outbound NSG rule to deny traffic to the internet.

Next, create the Azure virtual network peering from the spoke to the hub.

Configure client virtual machines

WSUS can be used to update any virtual machine that runs Windows. To set up clients using group policy, see [Configure client computers to receive updates from the WSUS server](#).

If you're an administrator managing a large network, see [Configure automatic updates and update service location](#) for information about how to use Group Policy settings to automatically configure clients.

Azure Update Manager

You can use the Azure Update Manager to manage and schedule operating system updates for VMs that are syncing against WSUS. The patch status of the VM (that is, which patches are missing) is assessed based on the source that the VM is configured to sync with. If the Windows VM is configured to report to WSUS, the results might differ from what Microsoft Update shows, depending on when WSUS last synced with Microsoft Update. After you configure your WSUS environment, you can enable Update Management. For more information, see [the overview of Azure Update Manager](#).

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Paul Reed](#) | Azure Compliance Senior Program Manager

Next steps

- For more information on planning a deployment, see [Plan your WSUS deployment](#).
- For more information on managing WSUS, setting up a WSUS synchronization schedule, and more, see [WSUS administration](#).

Related resource

- [Run a Windows VM on Azure](#)

Use the Azure Governance Visualizer to optimize cloud governance

Azure

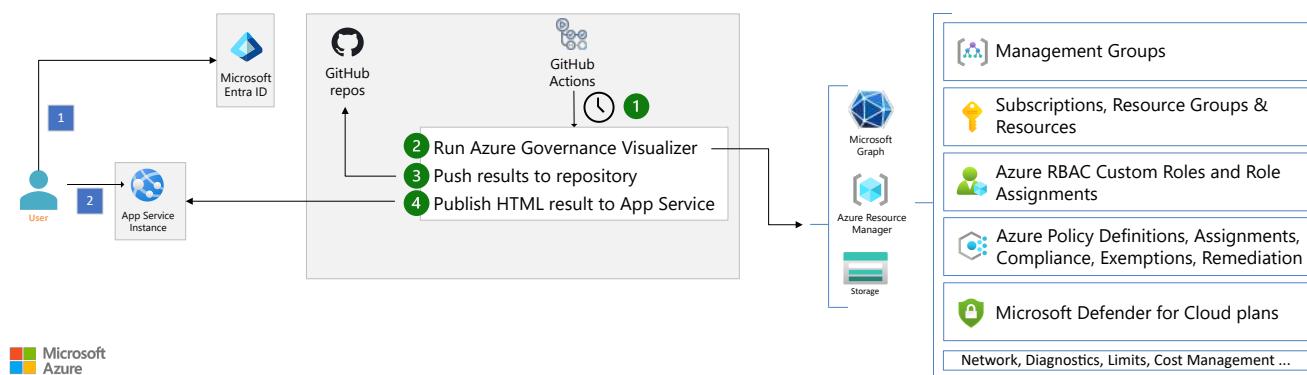
This article describes how to deploy the Azure Governance Visualizer. Organizations can use the Azure Governance Visualizer to capture pertinent governance information about their Azure tenants. The tool captures:

- Management group hierarchy.
- Policy information, such as custom policy definitions, orphaned custom policy definitions, and policy assignments.
- Azure role-based access control (Azure RBAC) information, such as custom role definitions, orphaned custom role definitions, and role assignments.
- Azure security and best practice analysis.
- Microsoft Entra ID insights.

The Azure Governance Visualizer should be automated through GitHub workflows. The visualizer outputs the summary as HTML, MD, and CSV files. Ideally, the generated HTML report is made easily accessible to authorized users in the organization. This article shows you how to automate running the Azure Governance Visualizer and host the reporting output securely and cost effectively on the Web Apps feature of Azure App Service.

An example implementation is available on GitHub at [Azure Governance Visualizer implementation](#) ↗.

Architecture



Download a [Visio file](#) ↗ of this architecture.

Data flow

The solution architecture implements the following workflow:

1. A timer triggers the GitHub Actions flow.
2. The flow makes an OpenID Connect connection to Azure. It then runs the Azure Governance Visualizer tool. The tool collects the required insights in the form of HTML, MD, and CSV reports.
3. The reports are pushed to the GitHub repository.
4. The HTML output of the Azure Governance Visualizer tool is published to App Service.

User flow

This flow explains how a user can use the tool:

1. The user browses to the App Service URL to access the HTML report of the visualizer. The user is required to authenticate through Microsoft Entra ID authorization.
2. The user can review the insights provided by the visualizer.

Components

The automation presented in this scenario consists of the following components:

- [Microsoft Entra ID](#) is an enterprise identity service that provides single sign-on, multifactor authentication, and other identity services to protect against cybersecurity threats. In this architecture, it's used to provide secure authentication and authorization to the Azure Governance Visualizer's web app to a specific Entra ID group.
- [Azure App Service](#) is a fully managed platform for creating and deploying cloud applications. It lets you define a set of compute resources for a web app to run, deploy web apps, and configure deployment slots. In this architecture, it's used to host the output of the Azure Governance Visualizer to provide secure and smooth access across the organization.
- [GitHub](#) is a popular SaaS offering from Microsoft that is frequently used by developers to build, ship, and maintain their software projects. In this architecture, it's used to host the infrastructure-as-code for the solution and the GitHub actions used to deploy and maintain it.
- [GitHub Actions](#) is a continuous integration and continuous delivery (CI/CD) platform that allows you to automate your build, test, and deployment pipeline. In this architecture, it provides continuous integration and continuous deployment capabilities to deploy and update the Azure Governance Visualizer.

Alternatives

The Azure Governance Visualizer is a PowerShell script, which can be run directly on a local machine. The visualizer can be configured to run as part of GitHub Actions to receive up-to-date information about your environment. The visualizer produces a wiki as an output that can be published in GitHub or Azure DevOps.

Scenario details

Azure Governance Visualizer is a PowerShell-based script that iterates your Azure Tenant's Management Group hierarchy down to subscription level. It captures most relevant Azure governance capabilities, such as Azure Policy, Azure RBAC, Microsoft Entra ID, and Blueprints. From the collected data, Azure Governance Visualizer visualizes all of this information in an easy to navigate HTML report.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that can be used to improve the quality of a workload. For more information, see [Microsoft Azure Well-Architected Framework](#).

Security

Security provides assurances against deliberate attacks and the abuse of your valuable data and systems. For more information, see [Overview of the security pillar](#).

Restricting the reporting HTML to only those users authorized to view this data is important. This data is a gold mine for both insider and external threats, as it exposes your Azure landscape, including security controls.

- Use Microsoft Entra authentication to restrict access to authorized individuals. Consider using Web Apps authentication to provide this service. The deployment code in GitHub configures Web Apps and actively verifies that authentication is enabled before deploying.
- Consider applying network security controls to expose the site to your team only over a [private endpoint](#). And to restrict traffic, consider using the IP restrictions of Web Apps.
- Enable access logging on the Azure web app to be able to audit access. Configure the Azure web app to send those logs to a Log Analytics workspace.

- Ensure secure communication is enabled on the Azure web app. Only HTTPS and FTPS are allowed, and the minimum version of TLS is configured as 1.2.
- Consider using [Microsoft Defender for Cloud's Microsoft Defender for App Service](#).
- Use the [latest versions of the runtime stack](#) of the Azure web app.
- Make sure to rotate the secret of this service principal regularly and monitor its activity. To gather all the required information, the visualizer deployed depends on a service principal with Microsoft Entra ID permissions.

For more information about security controls, see [Azure security baseline for App Service](#).

Cost optimization

Cost optimization is about looking at ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Overview of the cost optimization pillar](#).

- The B1 (Basic) tier is used for the deployed Azure web app in App Service. App Service hosts the HTML output of the Azure Governance Visualizer tool so it's lightweight. The visualizer can also be hosted on any other hosting platform that is secure and also cost-effective.
- Use the Azure pricing calculator to see a [pricing estimate for this solution](#).
- The sample in GitHub only deploys one instance of App Service, but you can choose to deploy more if needed.

Operational excellence

Operational excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Overview of the operational excellence pillar](#).

- The solution consists mainly of an Azure web app that hosts the HTML output of the visualizer tool. We recommend you enable the diagnostic settings of the web app to monitor traffic, access audit logs, metrics, and more.
- It's important to monitor the performance of the web app. Doing so helps to identify if you need to scale up or scale out depending on the amount of visualizer usage.
- It's also important to always run the [latest versions of the runtime stack](#) of the Azure web app.

- The Azure Governance Visualizer updates versions regularly with new features, bug fixes, or improvements. In the GitHub repository, a dedicated GitHub workflow handles the update process. There's a configurable option to update the visualizer's code automatically or manually by just opening a pull request with changes you can review and merge.
- The accelerate code might get updated with new settings on the App Service bicep code or with new instructions for the visualizer prerequisites. In the GitHub repository, a dedicated GitHub workflow handles this update process. There's a configurable option to update the visualizer's code automatically or manually by just opening a pull request with changes you can review and merge.

Deploy this scenario

To deploy this scenario, see the [Azure Governance Visualizer deployment GitHub repository](#).

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal authors:

- [Seif Bassem](#) | Cloud Solution Architect

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

- [Azure Governance Visualizer deployment GitHub repository](#)
- [Azure Governance Visualizer Open Source project](#)

Related resources

- [Azure landing zones - Bicep modules design considerations](#)
- [Azure landing zone overview](#)

Azure Arc hybrid management and deployment for Kubernetes clusters

Azure Arc

Azure Kubernetes Service (AKS)

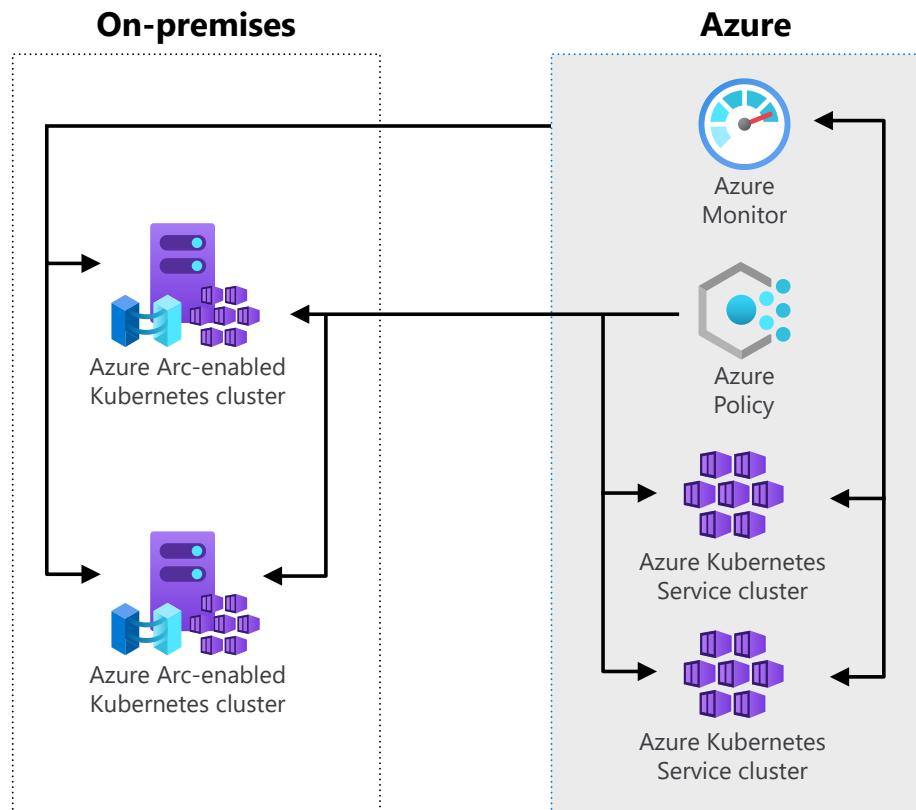
Azure Monitor

Azure Policy

Azure Role-based access control

This reference architecture describes how Azure Arc extends Kubernetes cluster management and configuration across customer datacenters, edge locations, and multiple cloud environments.

Architecture



Download a [Visio file](#) of this architecture.

Workflow

The following workflow corresponds to the previous diagram:

- **Azure Arc-enabled Kubernetes**: Attach and configure Kubernetes clusters inside or outside of Azure by using Azure Arc-enabled Kubernetes. When a Kubernetes cluster is attached to Azure Arc, it's assigned an Azure Resource Manager ID and a managed identity.
- **Azure Kubernetes Service (AKS)**: Host Kubernetes clusters in Azure to reduce the complexity and operational overhead of Kubernetes cluster management.
- **On-premises Kubernetes cluster** : Attach Cloud Native Computing Foundation (CNCF)-certified Kubernetes clusters that are hosted in on-premises or non-Microsoft cloud environments.
- **Azure Policy**: Deploy and manage policies for Azure Arc-enabled Kubernetes clusters.
- **Azure Monitor**: Observe and monitor Azure Arc-enabled Kubernetes clusters.

Components

- **Azure Arc** is a service that extends the Azure platform to enable building applications and services that can run across datacenters, at the edge, and in multicloud environments. In this architecture, Azure Arc serves as the foundational platform that enables centralized management and governance of Kubernetes clusters regardless of where they're hosted. It provides a unified control plane for hybrid and multicloud scenarios.
- **AKS** is a managed service for deploying and scaling Kubernetes clusters in Azure. In this architecture, AKS provides fully managed Kubernetes clusters within Azure. The clusters can be managed alongside on-premises and other cloud clusters through the same Azure Arc control plane, which reduces operational complexity.
- **Azure Policy** is a service that enables real-time cloud compliance at scale with consistent resource governance. In this architecture, Azure Policy provides centralized policy management and enforcement across all Arc-enabled Kubernetes clusters. It helps ensure consistent governance, security, and compliance policies whether clusters are running in Azure, on-premises, or in other clouds.
- **Azure Monitor** is a comprehensive monitoring solution that provides end-to-end observability for applications, infrastructure, and networks. In this architecture, Azure Monitor delivers unified monitoring and observability across all Kubernetes clusters in the hybrid environment. It collects metrics, logs, and performance data from both Azure-hosted and Azure Arc-enabled clusters for centralized analysis and alerting.

Scenario details

You can use Azure Arc to register Kubernetes clusters that are hosted outside of Microsoft Azure. You can then use Azure tools to manage these clusters and AKS-hosted clusters.

Potential use cases

Typical uses for this architecture include:

- Managing inventory, grouping, and tagging for on-premises Kubernetes clusters and AKS-hosted clusters.
- Using Azure Monitor to monitor Kubernetes clusters across hybrid environments.
- Using Azure Policy to help deploy and enforce policies for Kubernetes clusters across hybrid environments.
- Using Azure Policy to help deploy and enforce GitOps.
- Maximizing your on-premises graphics processing unit (GPU) investment by training and deploying Azure Machine Learning workflows.
- Using Azure Monitor managed service for Prometheus and Managed Grafana to monitor and visualize Kubernetes workloads.

Recommendations

You can apply the following recommendations to most scenarios. Follow these recommendations unless you have a specific requirement that overrides them.

Cluster registration

You can register any active CNCF Kubernetes cluster. You need a `kubeconfig` file to access the cluster and a cluster-admin role on the cluster to deploy Azure Arc-enabled Kubernetes agents. Use the Azure CLI to perform cluster registration tasks. The user or service principal that you use for the `az login` and `az connectedk8s connect` commands requires Read and Write permissions on the `Microsoft.Kubernetes/connectedClusters` resource type. The Kubernetes Cluster - Azure Arc Onboarding role has these permissions and can be used for role assignments on either the user principal or the service principal. Helm 3 is required to onboard the cluster that uses the `connectedk8s` extension. The Azure CLI version 2.3 or later is required to install the Azure Arc-enabled Kubernetes CLI extensions.

Azure Arc agents for Kubernetes

Azure Arc-enabled Kubernetes consists of a few agents (or *operators*) that run in the cluster that's deployed to the `azure-arc` namespace:

- The `deployment.apps/config-agent` watches the connected cluster for source control configuration resources that are applied on the cluster and updates the compliance state.
- The `deployment.apps/controller-manager` is an operator of operators that orchestrates interactions between Azure Arc components.
- The `deployment.apps/metrics-agent` collects metrics from other Azure Arc agents to ensure that these agents perform optimally.
- The `deployment.apps/cluster-metadata-operator` gathers cluster metadata, including the cluster version, node count, and Azure Arc agent version.
- The `deployment.apps/resource-sync-agent` synchronizes the previously mentioned cluster metadata to Azure.
- The `deployment.apps/clusteridentityoperator` maintains the Managed Service Identity certificate that's used by other agents to communicate with Azure.
- The `deployment.apps/flux-logs-agent` collects logs from the flux operators that are deployed as a part of source control configuration.
- The `deployment.apps/extension-manager` installs and manages the lifecycle of extension Helm charts.
- The `deployment.apps/kube-aad-proxy` handles authentication for requests sent to the cluster via the AKS cluster connect feature.
- The `deployment.apps/clusterconnect-agent` is a reverse proxy agent that enables the cluster connect feature to provide access to the API server of the cluster. It's an optional component that's deployed only if the cluster connect feature is enabled on the cluster.
- The `deployment.apps/guard` is an authentication and authorization webhook server that's used for Microsoft Entra role-based access control (RBAC). It's an optional component that's deployed only if Azure RBAC is enabled on the cluster.
- The `deployment.apps/extension-events-collector` collects logs related to extensions lifecycle management. It aggregates these logs into events that correspond to each operation, such as Create, Upgrade, and Delete.
- The `deployment.apps/logcollector` collects platform telemetry to help ensure the operational health of the platform.

For more information, see [Connect an existing Kubernetes cluster to Azure Arc](#).

Monitor clusters by using Azure Monitor container insights

Monitoring your containers is crucial. Azure Monitor container insights provides robust monitoring capabilities for AKS and AKS engine clusters. You can also configure Azure Monitor container insights to monitor Azure Arc-enabled Kubernetes clusters that are hosted outside of Azure. This configuration provides comprehensive monitoring of your Kubernetes clusters across Azure, on-premises, and in non-Microsoft cloud environments.

Azure Monitor container insights provides performance visibility by collecting memory and processor metrics from controllers, nodes, and containers. These metrics are available in Kubernetes through the Metrics API. Container logs are also collected. After you enable monitoring from Kubernetes clusters, a containerized version of the Log Analytics agent automatically collects metrics and logs. Metrics are written to the metrics store, and log data is written to the logs store that's associated with your Log Analytics workspace. For more information, see [Azure Monitor features for Kubernetes monitoring](#).

You can enable Azure Monitor container insights for one or more deployments of Kubernetes by using a PowerShell script or a Bash script.

For more information, see [Enable monitoring for Kubernetes clusters](#).

Use Azure Policy to enable GitOps-based application deployment

Use Azure Policy to make sure that each GitOps-enabled

`Microsoft.Kubernetes/connectedclusters` resource or

`Microsoft.ContainerService/managedClusters` resource has specific

`Microsoft.KubernetesConfiguration/fluxConfigurations` applied on it. For example, you can apply a baseline configuration to one or more clusters, or deploy specific applications to multiple clusters. To use Azure Policy, choose a definition from the [Azure Policy built-in definitions for Azure Arc-enabled Kubernetes](#) and then create a policy assignment. When you create the policy assignment, set the scope to an Azure resource group or subscription. Also set the parameters for the `fluxConfiguration` that's created. When the assignment is created, the Azure Policy engine identifies all `connectedCluster` or `managedCluster` resources that are in scope and then applies the `fluxConfiguration` to each resource.

If you use multiple source repositories for each cluster, such as one repository for the central IT or cluster operator and other repositories for application teams, activate this feature by using

multiple policy assignments and configure each policy assignment to use a different source repository.

For more information, see [Deploy applications consistently at scale by using Flux v2 configurations and Azure Policy](#).

Deploy applications by using GitOps

GitOps is the practice of defining the desired state of Kubernetes configurations, such as deployments and namespaces, in a source repository. This repository can be a Git or Helm repository, Buckets, or Azure Blob Storage. This process is followed by a polling and pull-based deployment of these configurations to the cluster by using an operator.

The connection between your cluster and one or more source repositories is enabled by deploying the `microsoft.flux` extension to your cluster. The `fluxConfiguration` resource properties represent where and how Kubernetes resources should flow from the source repository to your cluster. The `fluxConfiguration` data is stored encrypted at rest in an Azure Cosmos DB database to help ensure data confidentiality.

The `flux-config` agent that runs in your cluster monitors for new or updated `fluxConfiguration` extension resources on the Azure Arc-enabled Kubernetes resource, deploys applications from the source repository, and propagates all updates that are made to the `fluxConfiguration`. You can create multiple `fluxConfiguration` resources by using the `namespace` scope on the same Azure Arc-enabled Kubernetes cluster to achieve multi-tenancy.

The source repository can contain any valid Kubernetes resources, including Namespaces, ConfigMaps, Deployments, and DaemonSets. It can also contain Helm charts for deploying applications. Common source repository scenarios include defining a baseline configuration for your organization that can include common RBAC roles and bindings, monitoring agents, logging agents, and cluster-wide services.

You can also manage a larger collection of clusters that are deployed across heterogeneous environments. For example, you can have one repository that defines the baseline configuration for your organization, and then apply that configuration to multiple Kubernetes clusters simultaneously. You can also deploy applications to a cluster from multiple source repositories.

For more information, see [Deploy applications by using GitOps with Flux v2](#).

Run Machine Learning

In Machine Learning, you can choose an AKS (or Azure Arc-enabled Kubernetes) cluster as a compute target for your machine learning processes. This capability enables you to train or deploy machine learning models in your own, self-hosted (or multicloud) infrastructure. This approach allows you to combine your on-premises investments on GPUs with the ease of management that Machine Learning provides in the cloud.

Monitor Kubernetes workloads with managed Prometheus and Grafana

Azure Monitor provides a managed service for both Prometheus and Grafana deployments, so that you can take advantage of these popular Kubernetes monitoring tools. This managed service allows you to use these tools without the need to manage and update the deployments yourself. To analyze Prometheus' metrics, use the [metrics explorer with PromQL](#).

Topology, network, and routing

Azure Arc agents require the following protocols, ports, and outbound URLs to function.

Expand table

Endpoint (DNS)	Description
<code>https://management.azure.com:443</code>	Required for the agent to connect to Azure and register the cluster.
<code>https://[region].dp.kubernetesconfiguration.azure.com:443</code>	Data plane endpoint for the agent to push status and fetch configuration information, where [region] represents the Azure region that hosts the AKS instance.
<code>https://docker.io:443</code>	Required to pull container images.
<code>https://github.com:443, git://github.com:9418</code>	Example GitOps repos are hosted on GitHub. The configuration agent requires connectivity to the git endpoint that you specify.

Endpoint (DNS)	Description
<code>https://login.microsoftonline.com:443</code> , <code>https://<region>.login.microsoft.com</code> , <code>login.windows.net</code>	Required to fetch and update Azure Resource Manager tokens.
<code>https://mcr.microsoft.com:443</code> <code>https://*.data.mcr.microsoft.com:443</code>	Required to pull container images for Azure Arc agents.

For a complete list of URLs across Azure Arc services, see [Azure Arc network requirements](#).

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Microsoft Azure Well-Architected Framework](#).

Reliability

Reliability helps ensure that your application can meet the commitments that you make to your customers. For more information, see [Design review checklist for Reliability](#).

- In most scenarios, the location that you choose when you create the installation script should be the Azure region that's geographically closest to your on-premises resources. The rest of the data is stored within the Azure geography that contains the region you specify. This detail might affect your choice of region if you have data residency requirements. If an outage affects the Azure region that your machine is connected to, the outage doesn't affect the connected machine, but management operations that use Azure might not complete. If you have multiple locations that provide a geographically redundant service, connect the machines in each location to a different Azure region. This practice improves resiliency if a regional outage occurs. For more information, see [Supported regions for Azure Arc-enabled Kubernetes](#).
- You should ensure that the [services](#) in your solution are supported in the region where Azure Arc is deployed.

Security

Security provides assurances against deliberate attacks and the misuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

- You can use Azure RBAC to manage access to Azure Arc-enabled Kubernetes across Azure and on-premises environments that use Microsoft Entra identities. For more information, see [Use Azure RBAC for Kubernetes Authorization](#).
- Microsoft recommends that you use a service principal that has limited privileges to onboard Kubernetes clusters to Azure Arc. This practice is useful in continuous integration and continuous delivery pipelines such as Azure Pipelines and GitHub Actions. For more information, see [Create an Azure Arc-enabled onboarding service principal](#).
- To simplify service principal management, you can use managed identities in AKS. However, clusters must be created by using the managed identity. The existing clusters, which include Azure and on-premises clusters, can't be migrated to managed identities. For more information, see [Use a managed identity in AKS](#).

Cost Optimization

Cost Optimization focuses on ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

For general cost considerations, see [Cost Optimization design principles](#).

Operational Excellence

Operational Excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Design review checklist for Operational Excellence](#).

- Before you configure your Azure Arc-enabled Kubernetes clusters, review the Azure Resource Manager [subscription limits](#) and [resource group limits](#) to plan for the number of clusters.
- Use Helm, which is an open-source packaging tool, to install and manage the Kubernetes application lifecycles. Similar to Linux package managers such as APT and Yum, use Helm to manage Kubernetes *charts*, which are packages of preconfigured Kubernetes resources.

Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal author:

- [Pieter de Bruin](#) ↗ | Senior Program Manager

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

- [Azure Arc documentation](#)
- [Azure Arc-enabled Kubernetes documentation](#)
- [AKS documentation](#)
- [Azure Policy documentation](#)
- [Azure Monitor documentation](#)
- [Connect an existing Kubernetes cluster to Azure Arc](#)

Related resources

Related hybrid guidance:

- [Hybrid architecture design](#)
- [Azure hybrid options](#)

Related architectures:

- [Baseline architecture for AKS on Azure Local](#)
- [Optimize administration of SQL Server instances in on-premises and multicloud environments by using Azure Arc](#)

Hybrid availability and performance monitoring

Azure Event Hubs

Azure Log Analytics

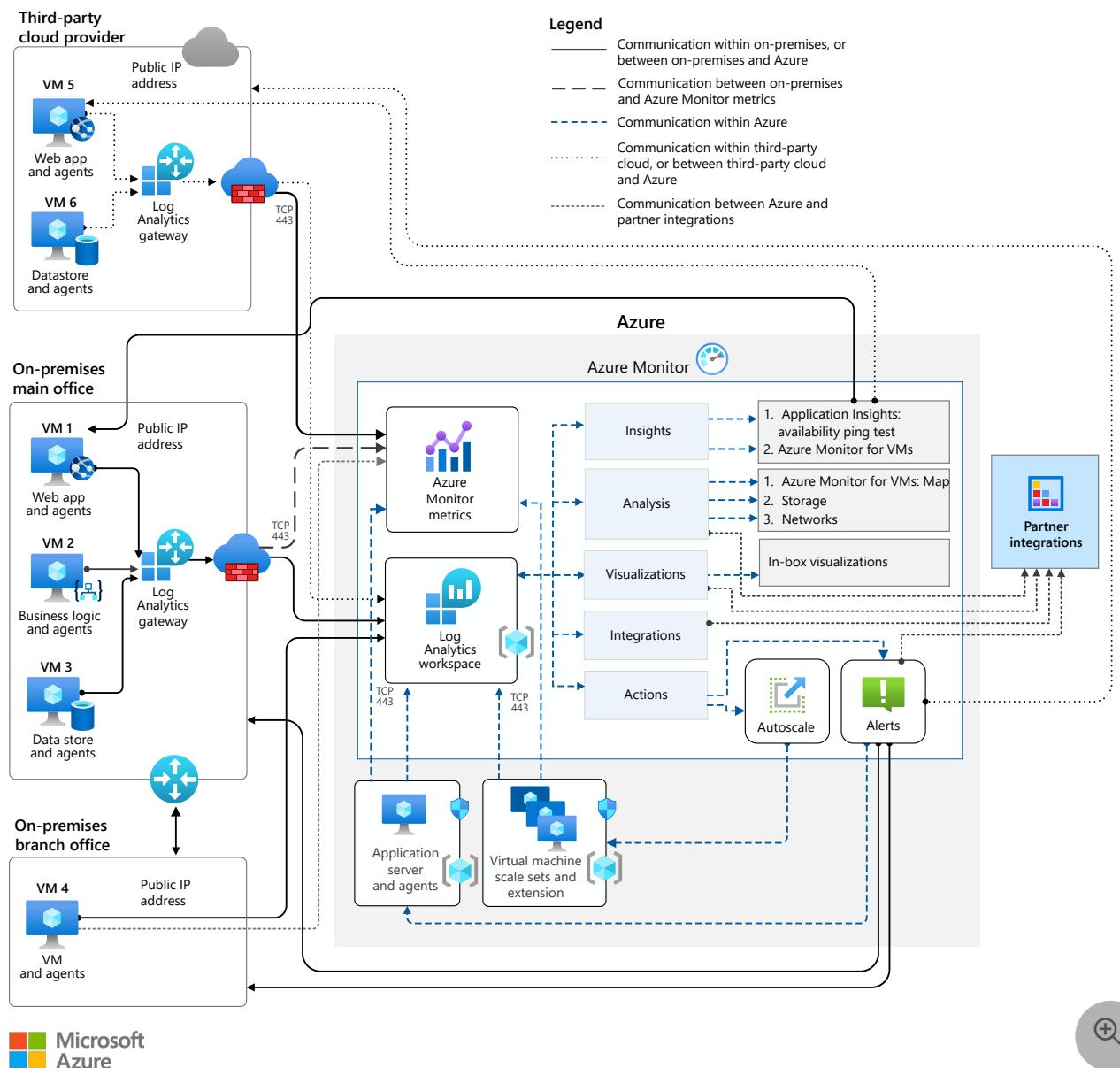
Azure Monitor

Azure Storage

Azure Virtual Machines

This reference architecture shows how to use Azure Monitor to monitor the performance and availability of operating system (OS) workloads that run in virtual machines (VMs). The VMs can be in Microsoft Azure, in on-premises environments, or in non-Azure clouds.

Architecture



Download a [Visio file](#) of this architecture.

Workflow

- **On-premises main office - VM 1.** This component is a web application with internet access and a public-facing webpage, and both Log Analytics and Dependency agents installed. For information about agents, refer to [Log Analytics agent overview](#) and [Overview of Azure Monitor agents, Dependency agent](#).
- **On-premises main office - VM 2.** This business-logic environment doesn't have internet access. It does, however, have Log Analytics and Dependency agents installed.
- **On-premises main office - VM 3.** This component is a datastore without internet access, but with Log Analytics and Dependency agents installed.
- **On-premises main office - Log Analytics gateway.** The Log Analytics gateway collects log and metric data from the three on-premises VMs, and delivers them into the *Log Analytics workspace* over Transmission Control Protocol (TCP) on port 443.
- **On-premises main office - Firewall.** Traffic to and from the on-premises environment is routed through the firewall.
- **Gateway.** The gateway provides connectivity to the branch office.
- **On-premises branch office - VM 4.** This component is the business application that's running without internet access, but with Log Analytics and Dependency agents installed. The Log Analytics agent installed on the VM is configured to transfer data directly to the Log Analytics workspace without the need for a Log Analytics gateway.
- **On-premises branch office - Gateway.** This gateway connects the branch office to the on-premises main office via a Virtual Private Network (VPN).
- **Third-party cloud provider - VM 5.** This component is a web application with internet access, a public-facing webpage, and both Log Analytics and Dependency agents installed.
- **Third-party cloud provider - VM 6.** This component is a datastore environment without internet access, but with both Log Analytics and Dependency agents installed. There is no direct connectivity from the third-party cloud provider environments to the on-premises environments.
- **Azure - VMSS.** This is a scale set that's created by using Azure Virtual Machine Scale Sets. It runs a business application with the log analytics and diagnostic agents installed.
- **Azure - Application server.** This server has a single VM running a business application, with Log Analytics and diagnostic agents installed.
- **Azure Monitor metrics.** Data collected by Azure Monitor metrics is stored in a time series database that's optimized for analyzing timestamped data. It also stores metrics sent from on-premises VMs and Azure VMs.
- **Azure Monitor - Log Analytics workspace.** This workspace stores logs sent from on-premises VMs, Azure VMs, and VMs on third-party cloud providers. The workspace is an Azure resource where data is aggregated, and it serves as an administrative boundary for accessing this data. Other Azure Monitor services then connect to the Log Analytics workspace and use the data for various purposes. For more information, see [Designing your Azure Monitor Logs deployment](#).

- **Azure Monitor - Insights - Application Insights.** Application Insights provides analyses of applications and insights into their use. In this example architecture, an availability ping test checks the availability of the on-premises web application. Alert rules are enabled to provide notification of a failed test. For more information, see [What is Application Insights?](#) and [Monitor the availability of any website](#).
- **Azure Monitor - Insights - Azure Monitor for VMs.** Azure Monitor for VMs monitors the performance and health of your virtual machines and virtual machine scale sets. The monitoring includes their running processes and dependencies on other resources. In this scenario, the Azure Monitor for VMs will provide insights into your virtual machines. For more information, see [What is Azure Monitor for VMs?](#).
- **Azure Monitor - Analysis.** Log and metric data from the VMs is queried within Azure Monitor metrics and the Log Analytics workspace using the Kusto Query Language (KQL). The results provide insights into the infrastructure, topology, and resources. For more information, see [Kusto: Overview](#) and [Azure Monitor log query examples](#).
- **Azure Monitor - Visualizations.** Azure Monitor uses visualization tools to review application and infrastructure components and communications between services in Azure Monitor. Visualization tools include [Application Map in Azure Application Insight](#), [the Map feature of Azure Monitor for VMs](#), [Azure Monitor Workbooks](#), and various dashboard views available within Azure Monitor. For more information, see [Use the Map feature of Azure Monitor for VMs to understand application components](#), [Create and share dashboards of Log Analytics data](#), and [Azure Monitor Workbooks](#).
- **Azure Monitor - Integrations.** Azure Monitor integrates with a range of partner and third-party tools and extensions. These tools and extensions enhance and build upon existing Azure Monitor functionality, such as analysis and visualizations.
- **Azure Monitor - Actions - Alerts.** Variations in metric and log data can indicate the occurrence of events. Rules define the data variations that trigger alerts, provide notifications, and initiate remediation responses. In this architecture, when an alert is triggered, automation runbooks automatically remediate the on-premises VMs and Azure VMs. Webhook actions, Service Management integration, and other action types are also available. For more information, see [Create, view, and manage metric alerts using Azure Monitor](#) and [Create, view, and manage log alerts using Azure Monitor](#).
- **Azure Monitor - Actions - Autoscale.** Autoscale adds or removes VM instances according to demand, which maintains performance and increases cost effectiveness. In this architecture, Autoscale has conditions defined around average CPU load (in percentage). When conditions are met, Azure Monitor Autoscale will adjust the scale set according to demand. For more information, see [Overview of autoscale in Microsoft Azure](#).

Components

The architecture consists of the following components:

- [Azure Event Hubs](#) is a real-time data ingestion service for streaming events. In this architecture, it connects Azure Monitor to external SIEM tools by streaming logs and metrics for advanced analytics and long-term retention.
- [Azure Monitor](#) is a unified platform for collecting and analyzing telemetry across environments. In this architecture, it serves as the central monitoring solution for performance, availability, and diagnostics across Azure, on-premises, and third-party cloud resources.
- [Azure Policy](#) is a governance tool for enforcing rules and automating resource configuration. In this architecture, it ensures consistent deployment of monitoring agents and enforces compliance across hybrid systems.
- [Azure Storage](#) is a cloud-based storage solution that supports blobs, files, queues, and tables. In this architecture, it retains monitoring data and diagnostic logs, providing scalable, durable, and secure storage for long-term retention and analysis.
- [Azure Virtual Machines](#) are scalable compute resources for running workloads in Azure. In this architecture, they host business applications and are monitored using Azure Monitor and diagnostic agents to ensure performance and availability.

Recommendations

The following best practices are recommendations that apply for most scenarios. Follow these practices unless you have a specific requirement that overrides them.

Log Analytics workspace

Consider the following recommendations when designing the Log Analytics workspace:

- Place the workspace and resources in the same Azure region, if latency is an important factor.
- Start with a single Log Analytics workspace, and increase the number of workspaces as the requirements change.
- If you have geographically dispersed teams and resources, you might need one workspace per region.
- Your workspace doesn't need to be in the same subscription as the resources you're running.

Alerts

For simpler scenarios, you can use metrics to flag alerts rather than logs. Metrics:

- Provide a count, or *numerical value*, for events such as CPU usage, available memory, or logical disk space.
- Have low latency.
- Offer greater granularity, for example per-second or per-minute intervals.
- Notify you about an issue quickly.

To collect custom performance indicators or business-specific metrics to provide deeper insights, use custom metrics. For more information, see [Custom metrics in Azure Monitor \(Preview\)](#).

Metrics alerts are not the answer in all situations. You might still want to use log-based alerts when you require more customization or more powerful correlations.

Analysis and Diagnostics

Consider the following recommendations for analysis and diagnostics:

- Use logs for deeper analysis. Logs can:
 - Provide verbose detail about events (compared to metrics).
 - Happen intermittently.
 - Facilitate deeper diagnostics after an initial metric flag.
- Customize log data collection (which is similar to metrics) using the HTTP Data Collector API to send log data to a Log Analytics workspace. For more information, see [Send log data to Azure Monitor with the HTTP Data Collector API \(public preview\)](#).
- Analyze your applications proactively with the **smart detection** feature of Application Insight. Smart detection applies the machine learning capabilities of Azure and statistical analysis to detect issues such as performance or failure anomalies, memory leaks, or general application degradation. For more information, see [Smart Detection in Application Insights](#).
- Use **Azure Monitor for VMs - Map** to review connections between servers, processes, inbound and outbound connection latency, and ports across any TCP-connected architecture. No configuration is required other than installing an agent. With **Azure Monitor for VMs - Map**, you can interact and engage with your servers as interconnected systems.

Log Analytics queries

Query the data within a **Log Analytics workspace** by using KQL to search for terms, specific events, or states to identify trends and analyze patterns. Use the **Query explorer** to browse and

select pre-written queries, modify them, or create your own. You can run, save, share, and export queries from within a workspace, and pin your favorite queries to a dashboard for reuse.

Agent installation

Install agents automatically and at scale, rather than individually, by using automation options such as Azure Policy, Azure PowerShell, Resource Manager templates, or Desired State Configuration (DSC). For more information, see [Enable Azure Monitor for VMs by using Azure Policy](#), [Enable Azure Monitor for VMs using Azure PowerShell](#), and [Enable Azure Monitor for VMs for a hybrid virtual machine - Desired State Configuration](#).

Dashboard

For critical applications, create an **Azure Dashboard** view. Share or make your dashboard available on a shared screen, in real time, to people who need critical application data.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that can be used to improve the quality of a workload. For more information, see [Microsoft Azure Well-Architected Framework](#).

Reliability

Reliability ensures your application can meet the commitments you make to your customers. For more information, see [Design review checklist for Reliability](#).

The following considerations help to ensure availability in your environment.

- Availability tests. The URL ping test used in this architecture is the simplest *outside-in* availability test. However, other options are available, such as:
 - Multi-step web test. Plays back recordings of sequenced web requests to test complex scenarios. Multiple-step web tests are created in Microsoft Visual Studio Enterprise, and then uploaded to the portal for execution.
 - Custom track availability tests. Use the `TrackAvailability()` method to send test results to Application Insights.
- Alerts. When you create an availability test in Application Insights, event alert notifications are enabled by default. You can edit the alert rules by specifying the notification type and details, from [Azure Monitor > Alerts](#).

Security

Security provides assurances against deliberate attacks and the abuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

The following items are considerations for making your environment more secure.

- Log Analytics workspace. Access modes are defined as one of the following contexts:
 - Workspace context. All logs that the workspace has permission to access can be queried. This is a vertical access approach. For example, a security team might need access to all resource data from the top down.
 - Resource context. Only logs for specific resources can be queried. For example, an application team can be granted access to logs for the particular resource they're working on.
- Secure data in transit to Log Analytics. Data in transit is secured using minimum Transport Layer Security (TLS) 1.2. You don't need to enable this feature explicitly. For more information, see [Log Analytics data security](#).
- Secure data at rest in Log Analytics. Data at rest in Log Analytics is secured, as per Azure Storage, using 256-bit Advanced Encryption Standard (AES) encryption by default.
- Smart Detection. Use Smart Detection in Application Insights to analyze the telemetry generated by your application, and to detect security issues. For more information, see [Application security detection pack \(preview\)](#).
- Integrate Azure Monitor with Security Information and Event Management (SIEM) tools. Route your monitoring data to an event hub with Azure Monitor to integrate external SIEM and monitoring tools. For more information, see [Stream Azure monitoring data to an event hub or external partner](#).

Cost Optimization

Cost Optimization is about looking at ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

The following items are considerations for controlling and managing costs in your environment.

- Azure Monitor. Azure Monitor costs are consumption-based, often referred to as *pay as you go*.
- Log Analytics. You pay for **data ingestion** and **data retention**. You can estimate and forecast the number of VMs, and the amount of data (in gigabytes) you expect to collect from each VM. A typical Azure VM consumes between 1 gigabyte (GB) and 3 GB of data

each month. If you're evaluating data usage with Azure Monitor logs, use the data statistics from your own environment and obtain a discount with **Capacity reservations**.

- Application Insights. This component is billed according to the volume of telemetry data your application sends, and the number of web tests you run.
- Metric queries. Metric queries are billed by the number of calls made.
- Alerts. Alerts are billed based on the type, and number, of signals monitored.
- Notifications. Notifications are billed according to the type, and number, of notifications you send.
- Azure Monitor. The **Usage and estimated costs** section of Azure Monitor estimates your monthly costs based on the previous 31 days of usage.
- For more information, see [Azure Monitor pricing](#) and [Pricing calculator](#).

Operational Excellence

Operational Excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Design review checklist for Operational Excellence](#).

Manageability

The following are considerations for making your environment more manageable.

- Azure Workbooks. Use workbooks to help perform further analysis, and create rich reports. Workbooks combine text, log queries, metrics, and parameters into interactive reports. Team members with access to the same Azure resources can edit workbooks. For more information, see [Create interactive reports Azure Monitor for VMs with workbooks](#).
- Partner integrations. Integrate Azure Monitor with partner and third-party tools to assist with analysis, visualization, alerts, or Service Management and Azure Pipelines. For more information, see [Azure Monitor partner integrations](#).
- Integrate Azure Monitor with Microsoft System Center. Integrate Azure Monitor with the System Center product suite. For more information, see [Connect Operations Manager to Azure Monitor](#).
- Send data to Azure Event Hubs. For integrating Azure Monitor with visualization and external monitoring tools, refer to [Stream Azure monitoring data to an event hub or external partner](#).
- Log Analytics gateway. For smaller environments such as the branch office, use the agent to transfer data into the Log Analytics workspace, rather than into a gateway. For more information, see [Establish connectivity to Azure Log Analytics](#).

DevOps

The following are considerations for integrating your environment with DevOps processes and solutions.

- Application Insights. Integrate Application Insights into Azure Pipelines to help make performance and usability improvements. Application Insights can detect performance anomalies automatically. It connects to various development tools, such as Azure DevOps Services and GitHub.
- Application Instrumentation. *Instrument* applications by modifying application code to enable telemetry with Application Insights. The following methods are ways to instrument applications:
 - At runtime. Instrumenting your web application on the server at runtime is ideal for applications that are deployed already, as it avoids having to update code. Suitable scenarios include:
 - Microsoft ASP.NET or ASP.NET Core applications hosted on Azure Web Apps
 - ASP.NET applications hosted in Microsoft Internet Information Services (IIS) on a virtual machine or virtual machine scale set
 - ASP.NET applications hosted in IIS on an on-premises VM
 - Java-based Azure Functions
 - Node.js apps on Linux App Services
 - Microservices hosted on AKS
 - At development time. Add Application Insights to your code to customize telemetry collection and send more data. Supported languages and platforms include:
 - ASP.NET applications
 - ASP.NET Core applications
 - .NET Console applications
 - Java
 - Node.js
 - Python
- Use IT Service Management Connector (ITSMC) to connect to external IT Service Management (ITSM) tools. ITSMC connects Azure to supported ITSM products and services, where issue-related work items typically reside. For more information, see [Connect Azure to ITSM tools using IT Service Management Connector](#).

Performance Efficiency

Performance Efficiency is the ability of your workload to meet the demands placed on it by users in an efficient manner. For more information, see [Design review checklist for Performance Efficiency](#).

The following are considerations for scaling your environment.

- Automate installation and configuration of your resources and applications.

- Large-scale geographically dispersed applications. Use **Distributed Tracing** within Application Insights to track dependencies and calls across multiple application components, backend resources, and microservices environments. With **Distributed Tracing** you can debug applications that call across process boundaries, outside the local stack. (You don't need to enable **Distributed Tracing**, it's available automatically as part of App Insights.)
 - Two options for consuming distributed trace data are:
 - Transaction diagnostics experience. This experience is similar to a call stack with an added time dimension. The transaction diagnostics experience provides visibility into one single transaction/request. It's helpful for finding the root cause of reliability issues and performance bottlenecks on a per-request basis. For more information, see [What is Distributed Tracing?](#)
 - Application map experience. This aggregates many transactions to demonstrate how systems interact topologically, and provide average performance and error rates. For more information, see [Application Map: Triage Distributed Applications](#).

Next steps

Learn more about the component technologies:

- [Azure Event Hubs — A big data streaming platform and event ingestion service](#)
- [Azure Monitor overview](#)
- [Overview of Log Analytics in Azure Monitor](#)
- [What are virtual machine scale sets?](#)
- [Overview of autoscale in Microsoft Azure](#)
- [What is Application Insights?](#)

Migration architecture design

Article • 06/20/2024

Azure provides resources for every stage of your cloud migration, with tools and guidance to help you move and manage your workloads.

These are just some of the key migration services available on Azure:

- [Azure Migrate](#). Simplify migration and modernization with a unified platform.
- [Azure Database Migration Service](#). Accelerate your data migration to Azure.
- [Azure Data Box](#). Easily move data to Azure when busy networks aren't an option.
- [Azure App Service migration tools](#). Quickly assess your web apps and migrate them to Azure with free, easy-to-use tools.

Introduction to migration on Azure

If you're new to migration on Azure, the best way to learn more is with [Microsoft Learn training](#), a free online training platform. Microsoft Learn provides interactive training for Microsoft products and more.

Here are some learning paths and modules to get you started:

- [Learning path: Best practices for Azure migration and modernization](#)
- [Learning path: Migrate virtual machines and apps using Azure Migrate](#)
- [Learning path: Migrate SQL workloads to Azure](#)
- [Module: Design migrations](#)
- [Module: Applications and infrastructure migration and modernization](#)
- [Module: Migrate to Azure through repeatable processes and common tools](#)

Path to production

For information about creating a migration plan, see [Build a migration plan with Azure Migrate](#).

Best practices

The Cloud Adoption Framework for Azure provides proven guidance and best practices that can help you confidently adopt the cloud and achieve business outcomes. Here are some migration best practices to check out:

- Azure cloud migration best practices checklist
- Multiple datacenters
- Azure regions decision guide
- Best practices when data requirements exceed network capacity during a migration effort
- Best practices to set up networking for workloads migrated to Azure
- Deploy a migration infrastructure
- Best practices to cost and size workloads migrated to Azure
- Scale a migration to Azure
- Governance or compliance strategy

For security best practices for Azure Migrate, see [Azure security baseline for Azure Migrate](#).

Migration architectures

The following sections provide links to reference architectures in a few high-level migration categories:

Hyper-V migrations

- Support matrix for Hyper-V migration
- How does Hyper-V replication work?

VMware migrations

- Support matrix for VMware migration
- Migrate workloads for Azure VMware Solution
- Azure Migrate agentless migration of VMware virtual machines
- Prepare for VMware agentless migration
- VMware agent-based migration architecture

Mainframe migrations

- Modernize mainframe and midrange data
- General mainframe refactor to Azure
- Rehost a general mainframe on Azure
- Migrate IBM mainframe applications to Azure with TmaxSoft OpenFrame

Oracle migrations

- Oracle database migration to Azure

Migrations of banking systems

- Banking system cloud transformation on Azure
- Patterns and implementations for a banking cloud transformation

Stay current with migration on Azure

Get the latest updates on [Azure migration services and features](#).

Additional resources

Example solutions

Following are some additional migration architectures to consider:

- Migrate an e-commerce solution to Azure
- Lift and shift to containers with AKS
- Migrate a monolithic application to microservices using domain-driven design
- Support matrix for migration of physical servers, AWS VMs, and GCP VMs
- Migrate a web app using Azure API Management

AWS or Google Cloud professionals

AWS

- [Azure Migrate](#) is comparable to [AWS Application Discovery Service](#). Azure Migrate assesses on-premises workloads for migration to Azure, performs performance-based sizing, and provides cost estimations.
- [Azure Database Migration Service](#) is comparable to [AWS Database Migration Service](#). Azure Database Migration Service enables seamless migrations from multiple database sources to Azure data platforms with minimal downtime.

Google Cloud

- [Google Cloud to Azure services comparison - Migration tools](#)

Feedback

Was this page helpful?

 Yes

 No

Why migrate from BizTalk Server to Azure Logic Apps?

07/03/2025

This guide provides an overview about the reasons and benefits, product comparisons, capabilities, and other information to help you start migrating from on-premises BizTalk Server to Azure Logic Apps. Following this guide, you'll find more guides that cover how to choose the services that best fit your scenario along with migration strategies, planning considerations, and best practices to help you deliver successful results.

Reasons and benefits

By migrating your integration workloads to Azure Logic Apps, you can reap the following primary benefits:

 [Expand table](#)

Benefit	Description
Modern integration platform as a service (iPaaS)	Azure Logic Apps is part of Azure Integration Services, which provides capabilities that didn't exist when BizTalk Server was originally built, for example: <ul style="list-style-type: none">- The capability to create and manage REST APIs- Scalable cloud infrastructure- Authentication schemes that are modern, more secure, and easier to implement- Simplified development tools, including many web browser-based experiences- Automatic platform updates and integration with other cloud-native services- Ability to run on premises (Azure Logic Apps hybrid deployment model)
BizTalk feature investments	Azure Logic Apps, the successor to BizTalk Server, includes some core BizTalk Server capabilities. For example, the Azure Logic Apps Rules Engine uses the same runtime as the BizTalk Business Rules Engine (BRE). To help you preserve customer investments in BizTalk Server, the workflow designer in Azure Logic Apps includes additional capabilities such as the Data Mapper tool when you use Visual Studio Code, support for running custom code, and native XML support.
Consumption-based pricing	With traditional middleware platforms, you must often make significant capital investments in procuring licenses and infrastructure, forcing you to "build for peak" and creating inefficiencies. Azure Integration Services provides multiple pricing models that generally let you pay for what you use. Although some pricing models enable and provide access to more advanced features, you have the flexibility to pay for what you consume.

Benefit	Description
Lower barrier to entry	BizTalk Server is a very capable middleware broker but requires significant time to learn and gain proficiency. Azure Logic Apps reduces the time required to start, learn, build, and deliver solutions. For example, Azure Logic Apps includes a visual designer that gives you a no-code or low-code experience for building the declarative workflows that you want to replace BizTalk orchestrations.
SaaS connectivity	With REST APIs becoming standard for application integration, more SaaS companies have adopted this approach for exchanging data. Microsoft has built an expansive and continually growing connector ecosystem with hundreds of APIs to work with Microsoft and non-Microsoft services, systems, and protocols. In Azure Logic Apps, you can use the workflow designer to select operations from these connectors, easily create and authenticate connections, and configure the operations they want to use. This capability speeds up development and provides more consistency when authenticating access to these services using OAuth2.
Multiple geographical deployments	Azure currently offers 60+ announced regions , more than any other cloud provider, so that you can easily choose the datacenters and regions that are right for you and your customers. This reach lets you deploy solutions in a consistent manner across many geographies and provides opportunities from both a scalability and redundancy perspective.

What is Azure Logic Apps?

Azure Logic Apps is a cloud-based and hybrid service for automating workflows and orchestrating business processes, applications, and data across hybrid environments by using a visual designer. This service is part of Azure Integration Services, which are a set of cloud-based, serverless, scalable, and Microsoft-managed building blocks for you to create comprehensive integration solutions and migrate existing BizTalk Server solutions:

[Expand table](#)

Service	Description
Azure Logic Apps	<p>Create and run automated logic app workflows that orchestrate your apps, data, services, and systems. You can quickly develop highly scalable integration solutions for your enterprise and business-to-business (B2B) scenarios. Use the visual workflow designer to orchestrate microservices, APIs, and line-of-business integrations. To increase scale and portability while automating business-critical workflows, deploy and run anywhere that Kubernetes can run.</p> <p>You can create either Consumption or Standard logic app resources. A Consumption logic app includes only one stateful workflow that runs in multitenant Azure Logic Apps. A Standard logic app can include multiple stateful or stateless workflows that run in single-tenant Azure Logic Apps, an App Service Environment v3, or on Azure Arc-enabled Kubernetes clusters (hybrid deployment model).</p>

Service	Description
	<p>For positioning Azure Logic Apps within Azure Integration Services, this guide focuses on Standard logic apps, which provide the best balance between enterprise features, cost, and agility. For more information, see Azure Logic Apps.</p>
Azure Functions	<p>Write less code, maintain less infrastructure, and save on costs to run applications. Without you having to deploy and maintain servers, the cloud infrastructure provides all the up-to-date resources needed to keep your applications running. For more information, see Azure Functions.</p>
Azure Data Factory	<p>Visually integrate all your data sources by using more than 90 built-in, maintenance-free connectors at no added cost. Easily construct Extract, Transform, and Load (ETL) and Extract, Load, and Transform (ELT) processes code-free in an intuitive environment, or you can write your own code. To unlock business insights, deliver your integrated data to Azure Synapse Analytics. For more information, see Azure Data Factory.</p>
Azure Service Bus	<p>Transfer data between applications and services, even when offline, as messages using this highly reliable enterprise message broker. Get more flexibility when brokering messages between client and server with structured first-in, first-out (FIFO) messaging, publish-subscribe capabilities, and asynchronous operations. For more information, see Azure Service Bus.</p>
Azure Event Grid	<p>Integrate applications using events delivered by an event broker to subscriber destinations, such as Azure services, other applications, or any endpoint where Event Grid has network access. Event sources can include other applications, SaaS services, and Azure services. For more information, see Azure Event Grid.</p>
Azure API Management	<p>Deploy API gateways side-by-side and optimize traffic flow with APIs hosted in Azure, other clouds, and on-premises. Meet security and compliance requirements, while you enjoy a unified management experience and full observability across all internal and external APIs. For more information, see Azure API Management.</p>



Logic Apps

Cloud and hybrid workflow orchestration service



API Management

Full API lifecycle management



API Center

Centralized API discovery hub



Event Grid

Event routing service



Service Bus

Reliable cloud message delivery



Functions

Event-driven serverless code



Data Factory

Data movement and ETL tool

Complementary Azure services

Beyond the previously described services, Microsoft also offers the following complementary services that provide underlying capabilities for Azure Integration Services and which you'll likely use in a migration project:

 [Expand table](#)

Service	Description
Azure Storage	<p>Provides highly available, massively scalable, durable, secure, and modern storage for various data objects in the cloud. You can access these data objects from anywhere in the world over HTTP or HTTPS using a REST API.</p> <p>Azure Integration Services uses these capabilities to securely store configuration and telemetry data for you while transactions flow through the platform. For more information, see Azure Storage.</p>
Azure role-based access control (Azure RBAC)	<p>Manage access to cloud resources, which is a critical function for any organization that uses the cloud. Azure RBAC is an authorization system built on Azure Resource Manager that provides fine-grained access management to Azure resources. You can manage who can access Azure resources, what they can do with those resources, and which areas they can access. For more information, see Azure RBAC.</p>
Azure Key Vault	<p>Provides capabilities to help you solve problems related to secrets management, key management, and certificate management.</p> <p>Azure Integration Services provides integration with Azure Key Vault through application configuration settings and through a connector. This capability lets you</p>

Service	Description
	<p>store secrets, credentials, keys, and certificates in a secure but convenient manner. For more information, see Azure Key Vault.</p>
Azure Policy	<p>Provides capabilities that help you enforce organizational standards and assess compliance in a scalable way. Through the compliance dashboard, you get an aggregated view so you can evaluate the overall state of the environment with the ability to drill down to per-resource, per-policy granularity.</p>
	<p>Azure Integration Services integrates with Azure Policy so you can efficiently implement widespread governance. For more information, see Azure Policy.</p>
Azure Networking	<p>Provides a wide variety of networking capabilities, including connectivity, application protection services, application delivery services, and networking monitoring.</p> <p>Azure Integration Services uses these capabilities to provide connectivity across services using virtual networks and private endpoints. For more information, see Azure Networking.</p>
Azure Event Hubs	<p>Build dynamic data pipelines and immediately respond to business challenges by streaming millions of events per second from any source with this fully managed, real-time data ingestion service that's simple, trusted, and scalable.</p> <p>API Management performs custom logging using Event Hubs, which is one of the best solutions when implementing a decoupled tracking solution in Azure. For more information, see Azure Event Hubs.</p>
Azure SQL Database	<p>At some point, you might need to create custom logging strategies or custom configurations to support your integration solutions. While SQL Server is commonly used on premises for this purpose, Azure SQL Database might offer a viable solution when migrating on-premises SQL Server databases to the cloud. For more information, see Azure SQL Database.</p>
Azure App Configuration	<p>Centrally manage application settings and feature flags. Modern programs, especially those running in a cloud, generally have many distributed components by nature. Spreading configuration settings across these components can lead to hard-to-troubleshoot errors during application deployment. With App Configuration, you can store all the settings for your application and secure their accesses in one place. For more information, see Azure App Configuration.</p>
Azure Monitor	<p>Application Insights, which is part of Azure Monitor, provides application performance management and monitoring for live apps. Store application telemetry and monitor the overall health of your integration platform. You also have the capability to set thresholds and get alerts when performance exceeds configured thresholds. For more information, see Application Insights.</p>
Azure Automation	<p>Automate your Azure management tasks and orchestrate actions across external systems within Azure. Built on PowerShell workflow so you can use this language's many capabilities. For more information, see Azure Automation.</p>

Supported developer experiences

This section describes the developer tools that BizTalk server and Azure Integration Services support:

 Expand table

Offering	Product or service with supported tools
BizTalk Server	<p>Each BizTalk Server version supports a specific version of Visual Studio.</p> <p>For example, BizTalk Server 2020 supports Visual Studio 2019 Enterprise or Professional. However, Visual Studio Community Edition isn't supported.</p>
Azure Integration Services	<ul style="list-style-type: none">- Azure Logic Apps (Standard): Azure portal and Visual Studio Code- Azure Logic Apps (Consumption): Azure portal and Visual Studio Code- Azure Functions: Azure portal, Visual Studio Code, and Visual Studio 2022- Azure API Management: Azure portal and Visual Studio Code- Azure Service Bus: Azure portal and Service Bus Explorer- Azure Data Factory: Azure portal and Visual Studio 2015

BizTalk Server versus Azure Logic Apps

To compare BizTalk Server with Azure Logic Apps and discuss how to migrate, let's first briefly summarize what BizTalk Server does. Originally available in 2000, BizTalk Server is an on-premises, stable, middleware platform that connects various systems by using adapters. This platform works as a broker between businesses, systems, or applications and is now a well-established integration platform. To simplify the challenge in combining different systems that are developed in different languages and can be connected using different protocols and formats, BizTalk Server offers the following main capabilities:

- Orchestration (business flow)

Provides the capability to create and run orchestrations or graphically defined business processes.

- Messaging

Provides the capability to communicate with a wide range of software applications. Adapters allow BizTalk Server's messaging component to interact with various protocols

and data formats.

The BizTalk Server engine includes the following components:

[+] [Expand table](#)

Component	Description
Business Rule Engine (BRE)	Evaluates complex sets of rules.
Enterprise Single Sign-On (SSO)	Provides the capability to map authentication information between Windows and non-Windows systems.
Business Activity Monitoring (BAM)	Enables information workers to monitor a running business process.
Group Hub	Enables support personnel to manage the engine and the orchestrations that run.

How does BizTalk Server work?

BizTalk Server uses a publish-subscribe messaging engine architecture with the [MessageBox database](#) at the heart. MessageBox is responsible for storing messages, message properties, subscriptions, orchestration states, tracking data, and other information.

When BizTalk Server receives a message, the server passes and processes the message through a pipeline. This step normalizes and publishes the message to MessageBox. BizTalk Server then evaluates any existing subscriptions and determines the message's intended recipient, based on the message context properties. Finally, BizTalk Server routes the message to the intended recipient, based on subscriptions or filters. This recipient is either an orchestration or a Send port, which is a destination to where BizTalk Server sends messages or source from where BizTalk Server can receive messages. BizTalk Server transmits messages through a Send port by passing them through a Send pipeline. The Send pipeline serializes the messages into the native format expected by the receiver before sending the messages through an adapter.

The MessageBox database has the following components:

- Messaging agent

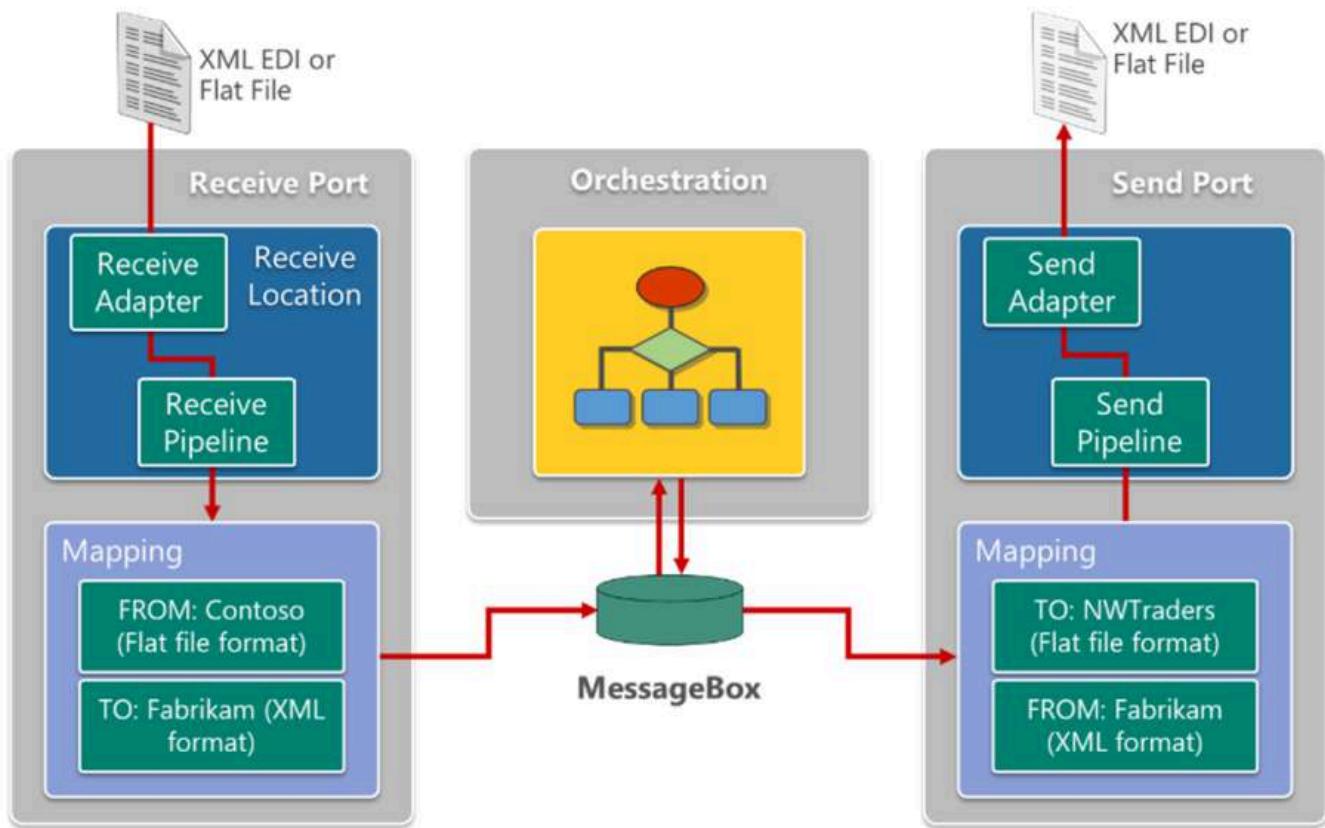
BizTalk Server interacts with MessageBox using this agent, which provides interfaces for publishing messages, subscribing to messages, retrieving messages, and more.

- One or more SQL Server databases

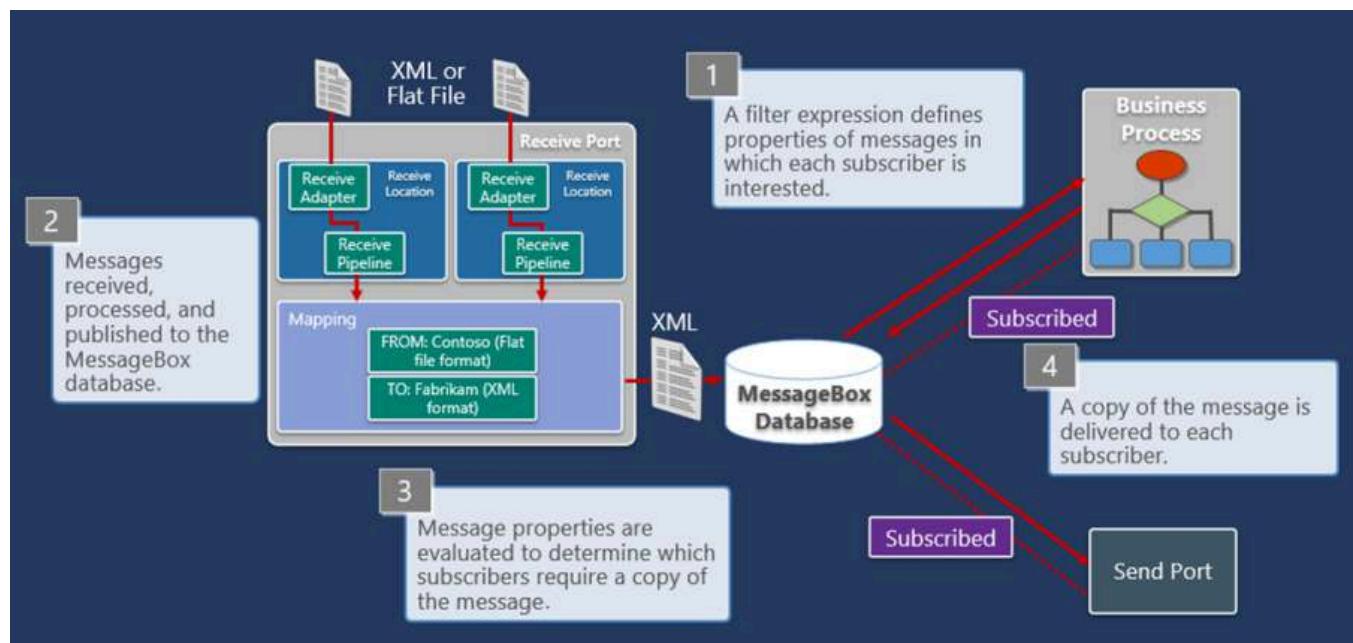
These databases provide the persistence store for messages, message parts, message properties, subscriptions, orchestration state, tracking data, host queues for routing, and

more.

The following image shows how BizTalk Server Messaging Engine works:



After a Receive port receives a message, MessageBox stores that message for processing by business processes or for routing to any Send ports that have subscriptions to specific messages.



For more information, see [Publish-subscribe architecture](#) later in this guide.

Business processes

This section describes options for designing and building business processes that you can run in BizTalk Server and Azure Integration Services.

BizTalk Server

In BizTalk Server, orchestrations are executable business processes that can subscribe to (receive) messages and publish (send) messages through the MessageBox database.

Orchestrations can construct new messages and can receive messages using the subscription and routing infrastructure. When MessageBox fills subscriptions for orchestrations, a new *instance* (orchestration run) activates, and MessageBox delivers the message. If necessary, the instance is rehydrated, and the message is then delivered. When messages are sent from an orchestration, they're published to MessageBox in the same way as a message arriving at a receive location with the appropriate properties added to the database for routing.

To enable publish-subscribe messaging, orchestrations use bindings that help create subscriptions. Orchestration ports are logical ports that describe an interaction. To deliver messages, you must bind these logical ports to a physical port, but this binding process is nothing more than configuring subscriptions for message routing.

BizTalk Server offers the following example advantages:

- Designer-first (declarative)

Design complex processes by using easy-to-understand design tools to implement patterns and workflows that might otherwise be difficult to implement in code.

- Abstraction with end systems

Design processes with focus on the messages, not the end system. For example, while developing your solutions, you don't have to worry about whether you're going to use a FILE adapter or an FTP adapter. Instead, you focus on the communication type, whether one way or request-response, and on the message type that you want to process. Later, when you deploy your solutions, you can then specify the adapter and the end systems.

Azure Logic Apps

In [Azure Logic Apps](#), you can create executable business processes and applications as logic app workflows by using a "building block" way of programming with a visual designer and prebuilt operations from hundreds of connectors, requiring minimal code. A logic app workflow starts with a trigger operation followed by one or more action operations with each operation functioning as a logical step in the workflow implementation process. Your workflow

can use actions to call external software, services, and systems. Some actions perform programming tasks, such as conditionals (if statements), loops, data operations, variable management, and more.

Azure Logic Apps offers the following example advantages:

- Designer-first (declarative)

Design complex processes by using easy-to-understand design tools to implement patterns and workflows that might otherwise be difficult to implement in code.

- Flexible and scalable

Azure Logic Apps is a cloud-based, serverless, highly scalable, computing service, which automatically scales and adapts to meet evolving business needs.

- Connects to anything

Select from a constantly expanding gallery with hundreds of prebuilt connectors to build your workflows. A connector provides operations that you can use as steps in your workflows. You can build integration solutions for most services and systems from both Microsoft and partners, including BizTalk Server, Salesforce, Office 365, SQL databases, most Azure services such as Azure Functions, Azure Storage, Azure Service Bus, and many others plus on-premises applications or systems, mainframes, midranges, SaaS, and APIs. If no prebuilt connector exists for the resource that you want to access, you can use the generic HTTP operation to communicate with the service, or you can create a custom connector.

Reusable components

Integration platforms offer ways to solve problems in a consistent and unified manner, which you can often achieve through reusable components. This section describes how you can reuse components in BizTalk Server and Azure Integration Services.

BizTalk Server

- Orchestrations

You can create and share common business logic as orchestrations across different workflows, internally inside the same application or with multiple applications. You can trigger orchestrations by using the native publish-subscribe mechanism in BizTalk Server (in a decoupled way) or by using the orchestration shapes named Call Orchestration for synchronous calls or Start Orchestration for asynchronous calls.

- Adapters

Adapters are software components that provide connectivity between BizTalk Server and trading partners using commonly recognized data protocols and document formats. These components make sending and receiving messages easier by using a delivery mechanism that conforms to a commonly recognized standard, such as SMTP, FTP, HTTP, and more. Adapters are part of the core platform, so all existing applications share them. You can also extend this layer by creating a custom adapter, either native or based on Windows Communication Foundation (WCF), by using the BizTalk Adapter Framework.

- Schemas

XML Schema Definition (XSD) schemas enable contract-based messaging in BizTalk Server. To avoid creating redundant schemas, you can reference schemas from compiled assemblies. To use shared schemas, you must add a reference to the shared assembly from your BizTalk project.

While this step might sound simple, managing changes to shared assemblies might become difficult due to dependency chaining. If the shared assembly requires an update, you must remove all projects that reference the shared assembly from BizTalk Server to install the update. However, to avoid these constraints, you can implement assembly versioning where you deploy a new version for a schema or shared schemas without breaking your existing solutions.

- Maps and custom functoids

Maps enable XML message translation or transformation in BizTalk Server. You can share maps, but like shared schemas, similar cautions apply to shared maps. Due to dependency chaining, proceed carefully and make sure that you have a mature software development lifecycle to manage change.

In maps, functoids perform calculations by using predefined formulas and specific values called arguments. BizTalk Server provides many functoids to support a range of diverse operations. Custom functoids provide a way for you to extend the range of available operations in the BizTalk Server mapping environment.

If you start to create many maps, you'll realize that you're repeatedly implementing similar logic. Consequently, you'll find yourself spending time maintaining multiple equivalent code snippets that you usually copy and paste into several locations within a map or across maps. Consider transforming such code snippets into a custom functoid. That way, you create the functoid only once, but you can reuse the functoid in as many maps as you want and update the functoid in only one place. Each custom functoid is deployed as a .NET assembly using classes derived from the

`Microsoft.BizTalk.BaseFunctoids` namespace. A single assembly can contain more than one custom functoid.

- .NET Framework assemblies

You can share these assemblies across BizTalk Server projects. These assemblies are easier to manage from a dependency perspective. Provided that no breaking changes exist, an update to a .NET Fx assembly requires updating the DLL in the Global Assembly Cache (GAC), which automatically makes the changes available to other assemblies. If breaking changes exist, you must also update the dependent project to accommodate the changes in the .NET Framework assembly.

- Custom pipelines and pipeline components

When BizTalk Server receives and sends messages, the server might need to prepare and transform messages for entry and exit, due to business reasons. In BizTalk Server, pipelines provide an implementation of the [Pipes and Filters integration pattern](#) and include many features such as a JSON encoder and decoder, MIME or SMIME decoder, and so on.

When you need to add information to the context of a message that requires pipeline customization, BizTalk Server provides the capability to customize these pipelines by creating custom pipeline components. A custom pipeline component is a .NET class that you use to implement multiple BizTalk interfaces and then use inside different stages of any custom pipeline. To write code for such a component, you can use C# or Visual Basic for .NET.

- Rules Engine policies

A Business Rules Engine policy is another kind of artifact that you can share across BizTalk Server applications deployed within the same [BizTalk group](#). If you have common Business Rules Engine rules, for example, related to message routing, you can manage these rules in one location and share them widely across installed BizTalk applications. The Business Rules Engine caches these rules, so if you make any updates to these rules, you must restart the Business Rules Engine Update Service. Otherwise, the changes are picked up in the next [Cache Timeout](#).

Azure Logic Apps

- Integration account

For Azure Logic Apps, an integration account is a cloud-based container and Azure resource that provides centralized access to reusable artifacts. For Consumption logic app

workflows, these artifacts include trading partners, agreements, XSD schemas, XSLT maps, Liquid template-based maps, certificates, batch configurations, and .NET Fx assemblies.

For Standard logic app workflows, Azure Logic Apps recently [introduced support for calling .NET Fx assemblies from XSLT transformations](#) without requiring an integration account. Alternatively, you can add schemas, maps, and assemblies to a Standard logic app project in Visual Studio Code and subsequently deploy to Azure.

- APIs

APIs enable digital experiences, make data and services reusable and universally accessible, simplify application integration, and underpin new digital products. With the proliferation and increasing dependency on APIs, organizations need to manage them as first-class assets throughout their lifecycle.

You can reuse APIs, especially those managed with Azure API Management, within Azure Logic Apps. After you add APIs to Azure API Management, you can use the API Management connector with logic app workflows to easily access APIs in a managed and governed manner. Azure Logic Apps also supports creating and using custom APIs so that your organization can promote reuse across the enterprise and avoid unnecessary redundant connectors that developers might otherwise create. Custom APIs also democratize who can use these APIs, rather than have developer figure out the mechanics to use a particular API.

- Custom connectors

If no prebuilt connectors exist for the APIs you want to use, you can wrap an external or external API with an OpenAPI schema to create a custom connector and access that connector from Consumption logic app workflows with the appropriate permissions. The custom connector creates a contract between Azure Logic Apps and the API that allows the easy assembly of request messages and for Azure Logic Apps to receive a typed response that you can use in downstream actions. Both REST APIs and SOAP APIs are supported, and they can reference either public APIs or private APIs that exist on your local network.

For Standard logic app workflows, you can create your own built-in custom connectors that are based on a service provider.

By implementing a custom connector, you simplify the development experience by creating a common interface for sending request messages and receiving typed responses. For more information, see [Custom connectors and APIs](#).

Adapters and connectors

The following section describes the concepts of adapters and connectors respectively in BizTalk Server and Azure Integration Services.

BizTalk Server

To exchange messages with external systems, applications, and entities, BizTalk Server provides adapters, which are COM or .NET Fx components that transfer messages to and from business endpoints such as file systems, databases, and custom business applications by using various communication protocols. BizTalk Server provides native adapters that support various protocols, for example:

- A File adapter that supports sending and receiving messages from a file location
- Adapters for EDI, FTP, HTTP, MSMQ, SMTP, POP3, and SOAP protocols
- An adapter for Windows SharePoint Services

The [BizTalk Adapter Framework](#) offers a stable, open mechanism for all adapters to implement or access work from the BizTalk Server Messaging Engine. The interfaces in the `Microsoft.BizTalk.Adapter.Framework` namespace enable adapters to modify configuration property pages. The BizTalk Adapter Framework also provides the capability to import services and schemas into a BizTalk project. Partner adapters are also available from various vendors and community members. For a list of known adapters, see [BizTalk Server: List of Third-Party Adapters](#).

Azure Logic Apps

When you build workflows with Azure Logic Apps, you can use prebuilt connectors to help you easily and quickly work with data, events, and resources in other apps, services, systems, protocols, and platforms, usually without having to write any code. Azure Logic Apps provides a constantly expanding gallery with hundreds of connectors that you can use. You can build integration solutions for many services and systems, cloud-based or on-premises, from both Microsoft and partners, such as BizTalk Server, Salesforce, Office 365, SQL databases, most Azure services, mainframes, APIs, and more. Some connectors provide operations that perform programming operations, such as conditional (if) statements, loops, data operations, variables management, and so on. If no connector is available for the resource that you want, you can use the generic HTTP operation to communicate with the service, or you can create a custom connector.

Technically, a connector is a proxy or a wrapper around an API that the underlying service or system uses to communicate with Azure Logic Apps. This connector provides the operations that you use in your workflows to perform tasks. An operation is available either as a trigger or action with properties that you can configure. Some triggers and actions also require that you

first create and configure a connection to the underlying service or system. If necessary, you'll also then authenticate access to a user account.

Most connectors in Azure Logic Apps are either a built-in connector or managed connector. Some connectors are available in both versions. The versions available depend on whether you create a Consumption logic app workflow or a Standard logic app workflow.

- Built-in connectors are designed to run natively on the Azure Logic Apps runtime and usually have better performance, throughput, capacity, or other benefits compared to any managed connector counterparts.
- Managed connectors are deployed, hosted, and managed by Microsoft in Azure. These connectors provide triggers and actions for cloud services, on-premises systems, or both. In Standard logic app workflows, all managed connectors are grouped as **Azure** connectors. However, in Consumption logic app workflows, managed connectors are grouped as either Standard or Enterprise, based on their pricing level.

For more information, see the following documentation:

- [Built-in connectors overview](#)
- [Managed connectors overview](#)
- [Managed connectors available in Azure Logic Apps](#)

Application connectivity

The following section describes options to connect with other applications from BizTalk Server and Azure Integration Services.

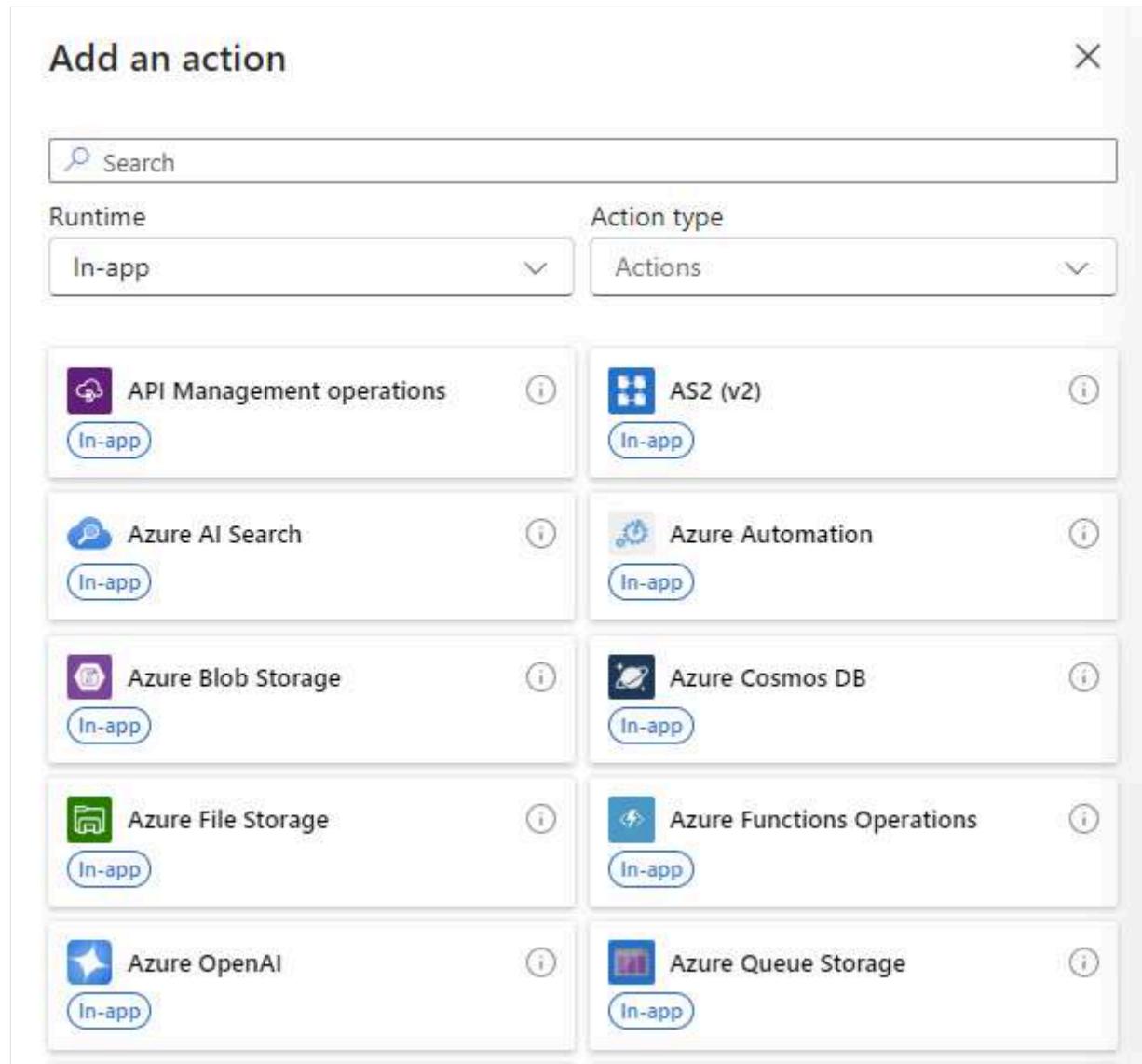
BizTalk Server

Adapters provide the connectivity capabilities in BizTalk Server and run locally on the BizTalk server that performs the send or receive operation. Approximately 30 out-of-the-box adapters are available, while a small ecosystem of ISV adapters provide additional functionality. With these adapters running locally, Windows Authentication is a popular authentication method. Commonly used adapters include FILE, SFTP, SQL, WCF (Basic-HTTP), HTTP, and SMTP. From this list, you can determine that the adapters in BizTalk Server are mostly protocol adapters. As a result, adapters usually use a message-oriented messaging pattern where a complete message is exchanged with other systems where those systems are responsible for parsing the data before loading the data into the final data store.

Azure Logic Apps

Connectors provide the connectivity capabilities in Azure Logic Apps and offer an abstraction on top of APIs that are usually owned by the underlying SaaS system. For example, services such as SharePoint are built using an API-first approach where APIs provide functionality to the service for end users, but the same functionality is exposed for other systems to call through an API. To simplify calling these APIs, connectors use metadata to describe the messaging contract so that developers know what data is expected in the request and in the response.

The following screenshot shows the connector operations search experience in the designer for a Standard logic app workflow in single-tenant Azure Logic Apps. When you select **Built-in**, you can find built-in connectors such as Azure Functions, Azure Service Bus, IBM DB2, SQL Server, Azure Storage, File System, HTTP, and more. If you select **Shared**, you can find over 1,000 connectors, including other Microsoft SaaS connectors, partner SaaS connectors, and so on.



The screenshot shows the 'Add an action' dialog in the Azure Logic Apps designer. At the top, there is a search bar labeled 'Search' and two dropdown menus: 'Runtime' set to 'In-app' and 'Action type' set to 'Actions'. Below these are five rows of connector cards, each with an 'In-app' button. The connectors listed are:

- API Management operations
- AS2 (v2)
- Azure AI Search
- Azure Automation
- Azure Blob Storage
- Azure Cosmos DB
- Azure File Storage
- Azure Functions Operations
- Azure OpenAI
- Azure Queue Storage

Web services and API connectivity

The following sections describe support for web services and API connectivity in BizTalk Server and Azure Logic Apps.

BizTalk Server

Web services support is a popular capability in BizTalk Server and is available by integrating with [Windows Communication Foundation \(WCF\)](#). This support in BizTalk falls into two categories: publishing and consuming WCF services.

[WCF adapters](#) provide the support for WS-* standards, such as WS-Addressing, WS-Security, and WS-AtomicTransaction. However, WS-ReliableMessaging isn't supported in this release of the WCF adapters.

WCF adapters support [Single Sign On \(SSO\)](#) through impersonation and acquire the Enterprise SSO ticket for using SSO with WCF adapters. This capability enables the user context to flow across systems. From an authentication perspective, [Service Authentication](#) supports the following types: None, Windows, and Certificate. [Client Authentication](#) supports the following types: Anonymous, UserName, Windows, and Certificate. Supported [security modes](#) include the following types: Transport, Message, and Mixed.

WCF supports transactions using the [WS-AtomicTransaction protocol](#), which you can find in WCF adapters such as WCF-WsHttp, WCF-NetTcp, and WCF-NetMsmq. This capability is supported in the following scenarios:

- Transactional submission of messages to the MessageBox database
- Transactional transmission of messages from the MessageBox to a transactional destination

The transactional scope is limited by the MessageBox component. For example, a BizTalk orchestration can't participate in a client's transaction. Similarly, a destination endpoint can't participate in a transaction that's initiated by a BizTalk orchestration.

WCF extensibility is available through WCF custom bindings. You'll need to compile and add custom code to the Global Assembly Cache (GAC). You'll also need to update the machine.config file to include the new extension. After the binding is installed, the extension is visible to the WCF-Custom and WCF-CustomIsolated adapters.

BizTalk Server can expose WCF-BasicHTTP receive locations as endpoints within Azure API Management when you use the BizTalk Administration Console. You can also expose your SOAP endpoints through API Management from BizTalk Server by using API Management in the Azure portal. For more information, see [Publish BizTalk WCF-BasicHTTP endpoints in API Management](#).

Azure Logic Apps

The connectivity model in Azure Logic Apps differs from BizTalk Server, partially due to the evolution of the API economy. As more organizations expose access to underlying systems and data, a platform-agnostic approach was needed. REST is now the dominant architectural approach to designing modern web services.

In Azure Logic Apps, [REST](#) is the default approach for connecting systems. As Microsoft and other software vendors expose RESTful services on top of their systems and data, Azure Logic Apps can expose and consume this type of information. The OpenAPI specification makes this capability possible for both humans and computers to understand the interaction between a client and server through metadata. As part of this understanding, both request and response payloads are derived, which means you can use dynamic content to populate a workflow action's inputs and use the outputs from the response in downstream actions.

Based on the software vendor who implements the underlying service that a connector calls, [authentication schemes](#) vary by connector. Generally, these schemes include the following types:

- [Basic](#)
- [Client certificate](#)
- [Active Directory OAuth](#)
- [Raw](#)
- [Managed Identity](#)

Microsoft provides strong layers of protection by [encrypting data during transit](#) and at rest. When Azure customer traffic moves between datacenters, outside physical boundaries that aren't controlled by Microsoft or on behalf of Microsoft, a data-link layer encryption method that uses [IEEE 802.1AE MAC Security Standards \(MACsec\)](#)  applies from point-to-point across the underlying network hardware.

Microsoft gives you the option to use [Transport Layer Security \(TLS\) protocol](#) for protecting data that travels between cloud services and customers. Microsoft datacenters negotiate a TLS connection with client systems that connect to Azure services. TLS provides strong authentication, message privacy, and integrity, which enables detection of message tampering, interception, and forgery along with interoperability, algorithm flexibility, and ease of deployment and use.

While this section focused on RESTful connectivity through connectors, you can implement SOAP web service connectivity through the custom connector experience or by using the API Management experience, which provides great SOAP capabilities. For more information, see [Increasing business value by integrating SOAP legacy assets with Azure logic Apps and Azure APIM](#) .

Block adapter or connector usage

The following sections describe options to prevent adapter or connector usage respectively in BizTalk Server and Azure Logic Apps.

BizTalk Server

BizTalk Server doesn't include the concept of blocking specific adapters from different applications, but you can "block" their usage in your applications by removing those adapters from the environment. Adapters in BizTalk Server are part of the platform settings, so installed adapters are available for anyone to use. You can also define specific receive and send handlers for each adapter, which defines the computers that belong to the [BizTalk group](#) that can execute or process those handlers.

Azure Logic Apps

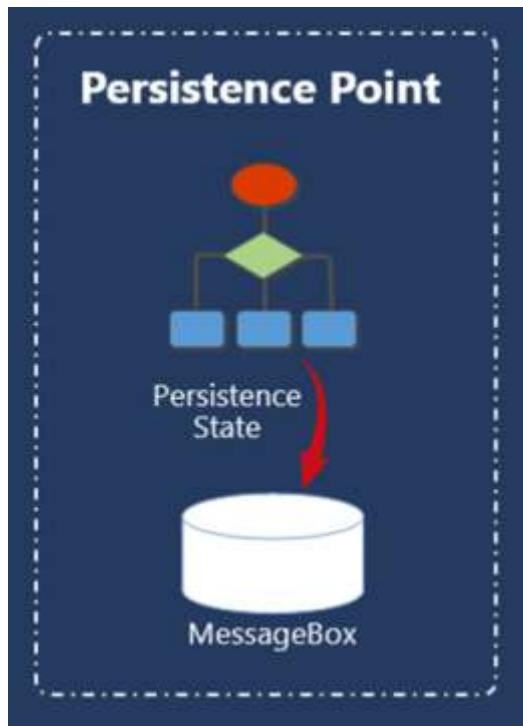
If your organization doesn't permit connecting to restricted or unapproved resources by using managed connectors in Azure Logic Apps, you can block the capability to create and use those connections in your logic app workflows. With Azure Policy, you can define and enforce policies that prevent creating or using the connections for connectors that you want to block. For example, for security reasons, you might want to block connections to specific social media platforms or other services and systems.

Message durability

The following section describes message persistence in BizTalk Server and Azure Integration Services.

BizTalk Server

The MessageBox database offers another benefit by acting as a persistence point that makes sure a message is persisted in storage before attempting to send to an endpoint. If the message fails to send after exhausting any configured retry attempts, the message is suspended and stored in MessageBox.



As an administrator, you can resume suspended messages from the BizTalk Administration Console. The same behavior happens when you use orchestrations. The Orchestration runtime persists the business logic, which you can resume if something goes wrong. For example, you can resume a message in an orchestration in the following scenarios:

- A message sent within a non-atomic scope
- At the end of a transactional scope
- When starting a new orchestration instance (Start Orchestration shape)
- In a debug breakpoint
- When the engine decides to dehydrate
- When the orchestration completes
- When system shuts down

BizTalk Server provides all these capabilities out-of-the-box. You don't need to worry about implementing persistence because BizTalk Server handles that for you.

Azure Logic Apps

Azure Logic Apps provides message durability in the following ways:

- Stateful workflows, which are the default in Consumption logic apps and available in Standard logic apps, have checkpoints that track the workflow state and store messages as they pass through workflow actions. This functionality provides access to rich data stored in the trigger and workflow instance run history where you can review detailed input and output values.

You can rerun a workflow instance either through Azure portal or an API. At this time, the entire workflow instance executes, regardless where any failure happened in the previous run. This behavior implies that messages are delivered *at least once*, and that idempotent processing happens at the consumers. You can also rerun the workflow instance starting from a specific action, currently in preview. This capability is available for all actions except for non-sequential and complex concurrency scenarios.

- With [peek-lock messaging](#) available in Azure Service Bus, you can either commit a message after successful message execution or abandon the message when a failure happens. To use this capability in Azure Logic Apps, select the Azure Service Bus connector. A committed message is removed from the message queue, while an abandoned message is unlocked and available for processing by clients. The peek-lock is a great way to achieve "exactly once" messaging.

Publish-subscribe architecture

The following sections describe options to implement the publish-subscribe pattern in BizTalk Server and Azure Logic Apps.

BizTalk Server

Publish-subscribe (pub-sub) capabilities exist through the [MessageBox database](#), which is described earlier in the section, [How does BizTalk Server work](#). A popular way to create subscriptions is by using Promoted Properties, which allow you to identify specific elements or attributes in a defined message schema as a Promoted Property. You can then establish subscriptions to filter messages based upon specific criteria against a Promoted Property. For example, if you promoted a schema element named City, you can then create a subscription that filters on the City element for specific cities. If your criteria are met, your subscription, a Send port, or an orchestration receives a copy of the message.

Azure Logic Apps

With an architecture completely different from BizTalk Server, most services in Azure Integration Services are event-based. Through [Azure Service Bus](#), Azure Logic Apps supports building publish-subscribe solutions. Azure Service Bus is a fully managed enterprise message broker with message queues and publish-subscribe topics in a namespace. You can use Azure Service Bus to decouple applications and services from each other, which provides the following benefits:

- Load balance work across competing workers.
- Safely route and transfer data with control across service and application boundaries.

- Coordinate transactional work that requires a high degree of reliability.

Azure Logic Apps includes an [Azure Service Bus connector](#) that you can use to publish and subscribe to messages. The benefit is that you can use messaging independently from your workflow. Unlike BizTalk Server, your messaging is decoupled from your workflow platform. Although messaging and workflow capabilities are decoupled in Azure Logic Apps, you can create message subscriptions in Azure Service Bus, which supports [message properties \(user properties\)](#). You can use these properties to provide key-value pairs that are evaluated by filters created on a [topic subscription](#). You define these user properties when you set up an Azure Service Bus operation by adding one or more key-value pairs. For a demonstration, see the following video: [Pub Sub Messaging using Azure Integration Services - Part 2 Content Based Routing](#).

Outside Azure Integration Services, you can also implement publish-subscribe scenarios by using we can also use [Azure Cache for Redis](#).

Business rules engine

The following section describes options for setting up business rules in BizTalk Server and Azure Integration Services.

BizTalk Server

BizTalk Server includes a forward-chaining rules engine that lets you construct "if-then-else" rules by using a visual editor. You can bundle these rules within a policy that's transportable to other environments in your IT landscape. These policies can also access XSD schemas, .NET Fx code, and SQL Server database tables to look up data and enrich outputs.

Azure Logic Apps

Azure Logic Apps includes the [Azure Logic Apps Rules Engine](#), currently in public preview. This rules engine includes the BizTalk Business Rules Engine (BRE) runtime so you can use reuse existing BizTalk BRE policies. Currently, support exists only for XML and .NET Framework facts.

Data transformation

The following sections describe the data transformation capabilities in BizTalk Server and Azure Logic Apps.

BizTalk Server

Provides rich tooling for you to transform XML messages from one format to another. Data transformation uses XSLT maps, which support extension objects that allow injecting custom .NET Fx code into the middle of these maps. You can also use out-of-the-box functoids that provide reusable functionality that helps you build rich maps.

Beyond the core XML transformations, BizTalk Server also provides encoding and decoding for CSV and JSON formats so you can convert between these formats and XML, giving you support for different formats.

Azure Logic Apps

- Enterprise Integration Pack

This component follows similar concepts in BizTalk Server and makes B2B capabilities easy to use in Azure Logic Apps. However, one major difference is that the Enterprise Integration Pack is architecturally based on integration accounts. These accounts simplify how you store, manage, and use artifacts, such as trading partners, agreements, maps (XSLT or Liquid templates), schemas, and certificates, for B2B scenarios.

- Liquid templates

For basic JSON transformations in logic app workflows, you can use built-in data operations, such as the Compose action or Parse JSON action. However, some scenarios might require advanced and complex transformations that include elements such as iterations, control flows, and variables. For transformations between JSON to JSON, JSON to text, XML to JSON, or XML to text, you can create a Liquid template that describes the required mapping or transformation using the Liquid open-source template language.

- XML operations

For XML transformations in logic app workflows, you can use built-in XML operations, such as the **Compose XML with schema** action and **Parse XML with schema** action.

- EDI schemas

EDI document schemas define the body of an EDI transaction document type. For your logic app workflows, all BizTalk EDI schemas in the [Microsoft Integration GitHub repository](#) are publicly available for you to use.

- Standard logic apps

In the Azure portal, you can upload maps and schemas directly to a Standard logic app resource. If you're working with a Standard logic app project in Visual Studio Code, you can upload these artifacts to their respective folders within the Artifacts folder without

using an integration account. You can also call custom compiled assemblies from XSLT maps.

- Azure Functions

You can execute XSLT or Liquid template transformations by using C# or any other programming language to create an Azure function that you can call with Azure API Management or Azure Logic Apps.

Network connectivity

The following section describes network connectivity functionality and capabilities in BizTalk Server and Azure Integration Services.

BizTalk Server

With BizTalk Server always installed in a server environment, network connectivity depends on the underlying server's network configuration. When you set up network connectivity for BizTalk Server, you usually have to configure the following areas:

- Dependencies
- Inbound and outbound connectivity to end systems

Dependencies configuration

To fully configure BizTalk Server in a multi-server environment, you need to pay special attention to all network connectivity dependencies, which usually involves firewall configuration to enable TCP and UDP ports for well-known services or protocols. For example, such services and protocols include access to a SQL Server engine, Microsoft Distributed Transaction Coordinator (MSDTC), clustered network drives, SSO services if installed on a different server, and SharePoint are all services that you must configure by creating inbound and outbound rules to implement connectivity.

Inbound and outbound connectivity configuration

After you fully set up BizTalk Server and are getting ready to deploy applications, make sure to implement firewall rules that allow host instances to connect and access different services, whether they're part of an internal or external network. When you're thinking about connectivity to end systems outside the organization network, you must also include security considerations. Various systems rely on defining a list of allowed IP addresses as their first line of defense, so ideally, BizTalk Server routes all their outbound communication through a well-defined list of public IP addresses.

When partner services try to contact BizTalk Server, make sure they don't reach an instance that's within your organization's network or inner layer where the core organization services might be available. Instead, give partner services access to an endpoint that exists within a perimeter network, also known as a demilitarized zone (DMZ), which is the outmost boundary of an organization's network. However, services to where BizTalk Server must route messages usually exist within your organization's network, so they should have access to that inner layer.

To achieve these scenarios, multiple approaches exist, for example:

- Implement BizTalk Server in a perimeter network and only allow its own services, or host instances, to access your organization's network
- Set up two BizTalk Servers with one in a perimeter network and the other in your organization's network. The server in the perimeter network then publishes the messages that the server in the organizational network consumes.
- Develop custom applications or appliance software, such as NetScaler and F5, which can act as reverse proxies, receive messages on behalf of BizTalk within the perimeter network, and redirect those calls to BizTalk Server.

Azure Logic Apps

- Inbound and outbound connectivity

Azure provides multiple ways to isolate their services within a network boundary and connect on-premises and cloud workloads. The following list describes different ways that you can integrate Azure resources with resources inside a network perimeter:

- On-premises data gateway

This gateway acts as a bridge between Azure and resources within a network perimeter, providing quick and secure data transfer between on-premises data and various Microsoft cloud services. These services include Azure Logic Apps, Microsoft Power BI, Microsoft Power Apps, Microsoft Power Automate, and Azure Analysis Services. With this gateway, you can keep databases and other data sources within their on-premises networks and securely use that on-premises data in cloud services.

- Hybrid Connections

Both an Azure service and a feature in Azure App Service, Hybrid Connections support scenarios and offers capabilities beyond those used in Azure App Service. For more information about usage outside Azure App Service, see [Azure Relay Hybrid Connections](#). Within Azure App Service, you can use Hybrid Connections to access application resources in any network that can make outbound calls to Azure over port 443. Hybrid Connections provide access from your app to a TCP endpoint and doesn't

enable a new way to access your app. In Azure App Service, each hybrid connection correlates to a single TCP host and port combination. This functionality enables your apps to access resources on any OS, provided that a TCP endpoint exists. Hybrid Connections doesn't know or care about the application protocol or what you want to access. This feature simply provides network access.

- Virtual network integration

With [Azure Virtual Network](#) integration, you can connect your Azure resource to a virtual network configured in Azure, giving your app access to resources in that virtual network. Virtual network integration in Azure Logic Apps is used only to make outbound calls from your Azure resource to your virtual network.

With [virtual network peering](#), you can connect your on-premises networks to Azure, which provides bi-directional connectivity between on-premises resources and Azure services. Azure Integration Services provides virtual network connectivity, allowing for hybrid integration. The following image shows a Standard logic app resource with the Networking page open and virtual network integration enabled as highlighted in the **Outbound Traffic** box. This configuration makes sure that all outbound traffic leaves from this virtual network.

The screenshot shows the Microsoft Azure portal interface for a StandardLogicApp. The left sidebar contains a navigation menu with various options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Microsoft Defender for Cloud, Events (preview), Workflows, Connections, Parameters, Artifacts, Schemas, Maps, Deployment, Deployment Center, Settings, Environment variables, Configuration, Authentication, Identity, Backups, Custom domains, Certificates, and Networking. The Networking option is highlighted with a red box. The main content area is titled 'StandardLogicApp | Networking' and contains sections for Inbound traffic configuration, Optional inbound services (Azure Front Door), and Outbound traffic configuration. The Outbound traffic configuration section shows 'Virtual network integration' set to 'FabrikamVNET/StandardLogicApp', which is also highlighted with a red box.

- Private endpoints

A **private endpoint** is a network interface that uses a private IP address from your virtual network. This network interface privately and securely connects to an Azure resource that's powered by [Azure Private Link](#). By enabling a private endpoint, you bring that Azure resource into your virtual network and allow resources in the network to make inbound calls to your Azure resource.

The following table shows the network connectivity methods that each Azure Integration Services resource can use:

[Expand table](#)

Resource	On-premises data gateway	Hybrid Connections	Virtual network integration	Private endpoints
Azure API Management	✓		✓	✓
Azure Logic Apps (Consumption)	✓			
Azure Logic Apps (Standard)	✓ (with Azure connectors)	✓ (with built-in connectors)	✓ (with built-in connectors)	✓
Azure Service Bus			✓	✓
Azure Event Grid				

Custom code

The following sections describe options for authoring and running your own code in BizTalk Server and Azure Logic Apps.

BizTalk Server

You can extend BizTalk in many ways by using custom .NET Fx code, for example:

[\[+\] Expand table](#)

Capability	Description
Inline code	You can write inline C# code within an Orchestration shape. You can also write inline code within a BizTalk Map. In both scenarios, the code snippets are generally simple in nature and can't be debugged.
Compiled assemblies	<p>You can call these assemblies from the following places:</p> <ul style="list-style-type: none"> - Expression shapes in an orchestration - BizTalk maps using the Scripting Functoid - Business Rules Engine policies - Pipelines as custom pipeline components <p>You can debug compiled assemblies by attaching the Visual Studio debugger to the appropriate host instance Windows process.</p>
Custom adapters	BizTalk Server includes many out-of-the-box adapters, but you can always create your own adapter if needed.

Capability	Description
Custom WCF behaviors	BizTalk Server includes many out-of-the-box adapters with the majority based on Windows Communication Foundation (WCF). In some cases, you might need to extend their capabilities by developing custom behaviors, such as applying an OAuth header to your system communication.
Extensibility in BizTalk Server maps	<ul style="list-style-type: none"> - You can create inline code using C#, JScript, Visual Basic, XSLT or XSLT Call Templates to suppress some limitations or difficulties using the out-of-the-box functoids. - You can call an external assembly using the Scripting Functoid. - You can create custom functoids to use across all your maps.

Azure Logic Apps

Azure Logic Apps provides the capability for you to author and run .NET code from your Standard logic app workflow. For this, you need to use [Visual Studio Code with the Azure Logic Apps \(Standard\) extension](#).

Also, the **Inline Code Operations** connector provides the actions named **Execute JavaScript Code**, **Execute CSharp Script Code (Preview)**, and **Execute PowerShell Code (Preview)**. You can use these actions to write small code snippets, which support dynamic content inputs and outputs. The Azure Logic Apps engine expects these code snippets to have short execution times. After a code snippet completes execution, the output is available for use by downstream actions in the workflow. Although no direct debugging support currently exists for this action, you can view the inputs and outputs in the workflow instance's run history.

As mentioned in the [Reusable Components](#) section, support for calling .NET Fx assemblies from an XSLT map is currently available in Consumption logic app workflows when you upload those assemblies to an integration account. This capability helps support custom data transformation rules. For Standard logic app workflows, the Azure Logic Apps team recently released support for calling .NET Fx code from XSLT maps without requiring an integration account. You can also add assemblies and maps to a Standard logic app project in Visual Studio Code and subsequently deploy to Azure. For more information, see [.NET Framework assembly support added to Azure Logic Apps \(Standard\) XSLT transformations](#) and the Road Map section.

You can also extend workflows by including Azure API apps or web apps created with Azure App Service. When you have a requirement to host web apps, REST APIs, and mobile backends, Azure App Service is the "go-to" HTTP-based solution. You can integrate apps hosted in Azure App Service with on-premises or cloud services. This platform supports both Windows and Linux-based environments to run and scale applications along with various languages and frameworks, such as ASP.NET Core, Java, Ruby, Node.js, PHP, and Python.

Application groups

The following sections describe options for organizing your workloads in BizTalk Server and Azure Logic Apps.

BizTalk Server

Part of your software development lifecycle includes building and managing your code and artifacts in logical packages. BizTalk Server supports the concept of an application such that you can deploy a Visual Studio solution into a BizTalk application. So, if you have scenarios where you need to share resources, you can reference other applications.

BizTalk Server uses an explicit sharing model where you can add references to compiled assemblies. Provided that these assemblies are in the Global Assembly Cache (GAC), the BizTalk Runtime finds and load assemblies as needed. One drawback is that when you need to update the shared assemblies, unless you implement a versioning scheme, you have to uninstall all BizTalk projects that reference your assemblies before you make your update. This limitation can result in lengthy deployment timelines and complexity in managing multiple installs and uninstalls.

Azure Logic Apps

In Azure Logic Apps, the Consumption logic app resource includes only a single stateful workflow, which means your workflow and logic app resource, which is your application, always has a 1-to-1 relationship. With the Standard logic app resource, the application concept evolved. Although your Standard logic app resource is still your application, you can include and run multiple workflows with this resource, resulting in a 1-to-many relationship. If you're working locally on a Standard logic app project in Visual Studio Code, your logic app resource maps to this single project. With this approach, you can easily and logically group related workloads, code, and artifacts in the same project and deploy that project as a single unit.

Cloud architectures work differently than server-based paradigms such as BizTalk. Azure Logic Apps (Standard) uses a pull model to bring in code and artifacts. So, as a result, you'll copy any additional necessary artifacts into your project and subsequently deploy them with your code and other artifacts. In some cases, you might want to avoid having to copy all the necessary code and artifacts. If so, you might consider turning this functionality into a service that you can manage separately but can call from a workflow.

For example, suppose you have a data transformation that's widely used by your organization. Rather than including the map for the transformation across multiple logic app projects, you can implement an interface that provides the transformation as a service. You can then manage

the lifecycle for that service separately from your logic app projects and call that service from your workflows.

With the ability to include multiple workflows in a Standard logic app project, you might ask how you'd organize those workflows within a project or across multiple projects? The answer usually depends on your requirements, for example:

- Business process affinity
- End to end monitoring and support
- Security, role-based access control, and network isolation
- Performance and business criticality
- Geo-location and geo-redundancy

For more information, see [Organizing logic app workflows in Azure Logic Apps \(Standard\)](#).

Security and governance

Security and governance are naturally important when building integrated solutions. By definition, middleware sits between two or more systems. To connect and access these systems when establishing a connection, you often need to pass along credentials or secrets, so managing this sensitive information requires consideration.

BizTalk Server

BizTalk includes [Enterprise Single Sign-On \(SSO\)](#), which lets you store, map and transmit encrypted credentials used by adapters. This encrypted information is stored in the SSO database. You can also configure [SSO affiliate applications](#), which are logical entities that represent a system or line of business system that you want to connect.

Azure Logic Apps

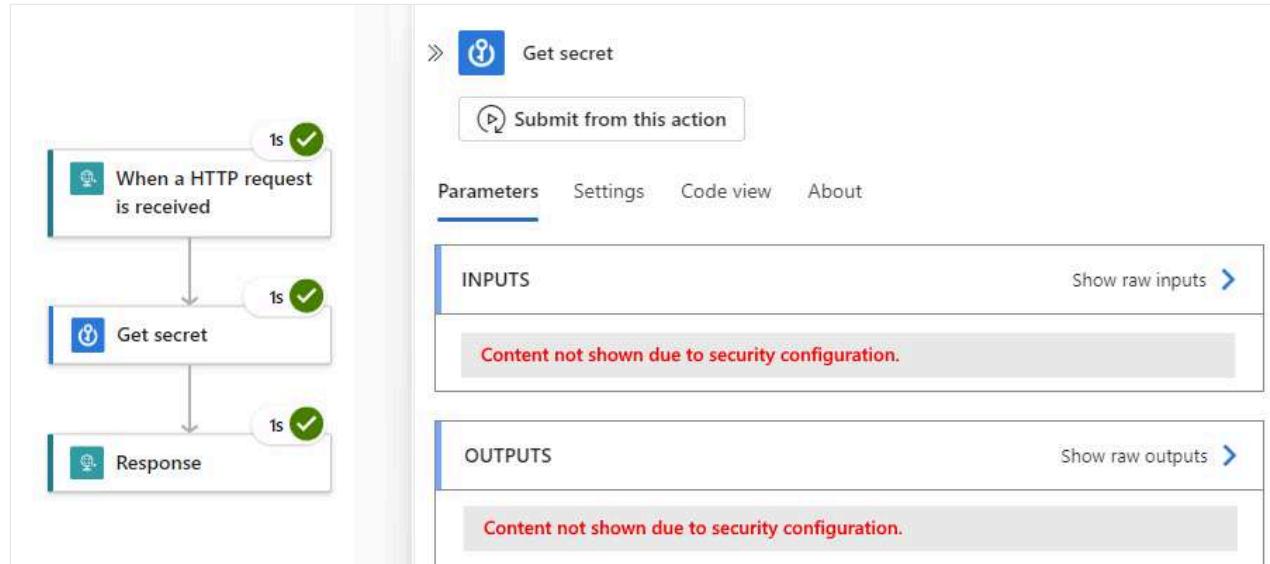
Azure Logic Apps supports the following security capabilities:

- Azure Key Vault

You can store credentials, secrets, API keys, and certificates using [Azure Key Vault](#). In Azure Logic Apps, you can access this information by using the [Azure Key Vault connector](#) and exclude this information from the platform's logs and run history by using the [secure inputs and outputs functionality](#).

Later in the [Tracking](#) section, this guide describes the run history functionality, which provides a step-by-step replay of a workflow's execution. Although Azure Logic Apps offers the value proposition of capturing every input and output in a workflow run,

sometimes you need to manage access to sensitive data more granularly. You can set up obfuscation for this data by using the secure inputs and outputs capability on triggers and actions to hide such content from run history and prevent sending this data to Azure Monitor, specifically Log Analytics and Application Insights. The following image shows an example result from enabling secure inputs and secure outputs in run history.



- OAuth-based integration

Most connectors use this authentication type when creating connections. This approach makes integrating with many SaaS services as easy as providing your email address and password. Azure API Management also supports OAuth, so you can use both services together by providing a unified authentication scheme.

This capability isn't natively available in BizTalk Server.

- Managed identities

Azure Logic Apps (Standard) can authenticate access to storage accounts by using a [managed identity](#). Also, some connectors support using managed identities for authenticating access to resources protected by Microsoft Entra ID. When you use a managed identity to authenticate your connection, you don't have to provide credentials, secrets, or Microsoft Entra tokens.

Application management and access management

The following section describes options for managing applications and access in BizTalk Server and Azure Integration Services.

BizTalk Server

Administrators use the [BizTalk Server Administrator Console](#) to manage BizTalk Server applications. This tool is a Microsoft Management Console (MMC) thick client application that administrators can use to deploy applications, review previous, active, and queued transactions, and perform deep troubleshooting activities such as reviewing traces and resubmitting transactions.

Azure Logic Apps

The [Azure portal](#) is a common tool that administrators and support personnel use to view and monitor the health of interfaces. For Azure Logic Apps, this experience includes rich transaction traces that are available through run history.

Granular [role-based access controls \(RBAC\)](#) are also available so you can manage and restrict access to Azure resources at various levels.

Storage

The following section describes options for data storage in BizTalk Server and Azure Integration Services.

BizTalk Server

BizTalk Server heavily relies on [SQL Server for data store and data persistence](#). All other components and hosts in BizTalk Server have specific roles in integrating disparate business applications, such as receiving, processing, or routing messages. However, the database computer captures and persists this work to disk. For example, when BizTalk Server receives an incoming message, the receive host persists that message to the MessageBox database before other hosts retrieve the message for orchestration processing and sending.

As you're responsible for provisioning and managing your SQL databases, high availability is an important architectural component to ensure uptime. To provide high availability for the BizTalk Server databases, customers often use Windows Clustering to create a server cluster with two or more computers running SQL Server. This server cluster provides redundancy and fault tolerance for BizTalk Server databases. Unlike load-balanced clustering where a computer group works together to increase availability and scalability, server clustering typically involves a pair of database computers in an active-passive configuration so that one computer provides backup resources for the other.

Azure Logic Apps

Azure Logic Apps relies on [Azure Storage](#) to store and automatically [encrypt data at rest](#). This encryption protects your data and helps you meet your organizational security and compliance commitments. By default, Azure Storage uses Microsoft-managed keys to encrypt your data. For more information, see [Azure Storage encryption for data at rest](#).

When you work with Azure Storage through the Azure portal, [all transactions take place over HTTPS](#). You can also work with Azure Storage by using the Storage REST API over HTTPS. To enforce using HTTPS when you call the REST APIs to access objects in storage accounts, enable the secure transfer that's required for the storage account.

Data configuration

The separation between configuration and code becomes important when you want to move your integration solutions between environments without having to recompile or reassemble your code. Configuration information is usually environment specific, so you can define endpoints and other details that need to change as you deploy solutions across your landscape.

BizTalk Server

- BizTalk NT Service executable

This executable calls an app.config file named [BTSNTSvc.exe.config](#). This file provides key-value pairs so that you can store clear text configuration information. However, take care with this file based on the following considerations:

- Make sure to carefully replicate configuration across all computers within a [BizTalk group](#).
- Configuration changes require that you restart host instances to pick up the latest values in this configuration file.
- Any syntax errors introduced to this configuration file prevent host instances from starting and result in downtime.

- Enterprise SSO tool

You can also use this tool as a configuration store. [Community tools](#) are also available to enable data management using Enterprise SSO. You can subsequently access this data through SDK tooling to retrieve this data at runtime.

- Custom cache components

These components are often introduced so you can address use cases beyond key-value pairs. For example, suppose you want to store tabular data in a SQL Server database and load that data into memory when a host instance starts. This implementation allows BizTalk Server to get this information at runtime by running custom .NET Fx code. You can then access this data from orchestrations, BizTalk maps, and custom pipeline components.

- Custom database

Databases are a well-known technology and language for both developers and administrators, so a custom database is another common option for storing application configuration data.

- Business Rules Engine (BRE)

While not a primary use case, the BRE can also act as a configuration store. Whether you call the engine from an orchestration or [pipeline component](#), you can define environment-specific information in BRE policies and then deploy the corresponding policy to the relevant environment. At runtime, an orchestration or pipeline component can access and use this information in downstream functions such as maps or in routing situations.

- Custom configuration file

You can use custom configuration (.config) files to store application configuration data, but this approach isn't common because you likely have to maintain a static and fixed location for these files across all environments.

- Windows registry

You can use the [Windows registry](#) as a valid option to store the application configuration values. This registry is a central hierarchical database used by Microsoft Windows operating systems to store information that's necessary to configure the system for one or more users, applications, and hardware devices. The registry contains the following basic elements: hives, keys and values. However, maintaining values stored in the registry can prove difficult in large environments with multiple registries and the difficulty of backing up individual application settings.

Azure Logic Apps

- Azure Key Vault

This service stores and protects cryptographic keys and other secrets used by applications and cloud services. Because secure key management is essential to protect data in the cloud, use [Azure Key Vault](#) to encrypt and store keys and secrets, such as passwords.

- Azure App Configuration

This service centrally manages application settings and feature flags. You can store configurations for all your Azure apps in a universal, hosted location. Manage configurations effectively and reliably in real time and without affecting customers by avoiding time-consuming redeployments. [Azure App Configuration](#) is built for speed, scalability, and security.

- Azure Cosmos DB

This service is a fully managed NoSQL database for modern app development with single-digit millisecond response times plus automatic and instant scalability that guarantee speed at any scale. You can load configuration data into [Azure Cosmos DB](#) and then access that data using the [Azure Cosmos DB connector](#) in Azure Logic Apps.

- Azure Table Storage

This service provides another storage facility to keep configuration data at a low cost. You can easily access this data using the [Azure Table Storage connector](#) in Azure Logic Apps. For more information, see [Azure Table Storage](#).

- Custom caching

You can also implement custom caching solutions with Azure Integration Services. Popular approaches include using [caching policies](#) in [Azure API Management](#) and [Azure Cache for Redis](#).

- Custom database

Databases are a well-known technology and language for both developers and administrators, so a custom database is another common option for storing application configuration data.

Large file processing

The following section describes options for handling large files in BizTalk Server and Azure Integration Services.

BizTalk Server

To address large file processing, BizTalk Server includes [optimizations](#) based upon the following profiles:

- Message routing only

If you use BizTalk Server only for routing messages based on promoted message properties, messages are streamed to the MessageBox database using the .NET XmlReader interface. BizTalk Server doesn't load individual message parts into memory, so in this scenario, out of memory errors aren't a problem. However, the primary consideration is the amount of time required to write very large messages (over 100 MB) to the MessageBox database. The BizTalk Server development team has successfully tested the processing of messages up to 1 GB when only performing routing. For more information, see [Optimizing pipeline performance](#).

- Data transforms with maps

When BizTalk Server transforms a document using a map, this potentially memory-intensive operation passes the message to the .NET XslCompiledTransform class, which loads the XSL style sheet. After the load operation successfully completes, multiple threads can simultaneously call the [Transform method](#). For more information, see [XslCompiledTransform Class](#).

BizTalk Server significantly improves memory management for large documents by implementing a configurable message size threshold for loading documents into memory during transforms. By default, the message size threshold is 1 MB. For any message with a size below this threshold, BizTalk Server handles the message within memory. To reduce memory requirements for any message with a size above this threshold, BizTalk Server buffers the message to the file system.

Azure Logic Apps

Some foundational differences exist between processing large files with an on-premises middleware platform such as BizTalk Server versus a PaaS offering such as Azure Logic Apps. For example, carefully scrutinize large message scenarios to find the right solution because potentially different ways exist to solve this problem in a modern cloud environment.

File size limits

In Azure, file size limits exist to ensure consistent and reliable experiences. To validate your scenario, make sure to review the [service limits documentation for Azure Logic Apps](#). Some connectors support [message chunking](#) for messages that exceed the default message size limit, which varies based on the connector. Message chunking works by splitting a large message into smaller messages.

Azure Logic Apps isn't the only service that has message size limits. For example, Azure Service Bus also has [such limits](#). For more information about handling large messages in Azure Service Bus, see [Large messages support](#).

Claim-check pattern

To avoid file size limitations, you can implement the [claim-check pattern](#), which works by splitting a large message into a *claim check* and a payload. You send the claim check to the messaging platform and store the payload on an external service. That way, you can process large messages, while you protect the message bus and the client from overload. This pattern also helps to reduce costs because storage is usually cheaper than resource units used by the messaging platform.

Azure Data Factory

[Azure Data Factory](#) provides another option for handling large files. This service is Azure's [ELT offering](#) for scalable serverless data integration and data transformation with a code-free visual experience for intuitive authoring and single-pane-of-glass monitoring and management. You can also lift and shift existing [SQL Server Integration Services \(SSIS\)](#) packages to Azure and run them with full compatibility in Azure Data Factory. The SSIS Integration Runtime offers a fully managed service, so you don't have to worry about infrastructure management. For more information, see [Lift and shift SQL Server Integration Services workloads to the cloud](#).

In on-premises architectures, SSIS was a popular option for managing the loading of large files into databases. As the cloud equivalent for that architecture, Azure Data Factory can address the transformation and movement of large datasets across various data sources, such as file systems, databases, SAP, Azure Blob Storage, Azure Data Explorer, Oracle, DB2, Amazon RDS, and more. When you have large data processing requirements, consider using Azure Data Factory as a better option over Azure Logic Apps and Azure Service Bus.

Monitoring and alerts

BizTalk Server

- [BizTalk Health Monitor](#)

This tool is an MMC snap-in that you can use to monitor the health of your BizTalk Server environments and perform maintenance tasks. Features include [MsgBox Viewer \(MBV\)](#) reports, [Terminator tool](#) tasks, email notifications, report collection, and [perfmon](#) integration.

- [BizTalk Administration console](#)

This tool is also an MMC snap-in for administrators to discover failures, suspended instances, transactions currently being retried, state, and more. The tool experience is very

reactive in nature because you must constantly refresh the console to review the latest information.

- [BizTalk360](#)

An external web solution that provides total control over your BizTalk Server environment. This single tool offers operations, monitoring, and analytics capabilities for BizTalk Server.

Azure Logic Apps

In Azure Logic Apps, the following options are available:

- For Consumption logic app workflows, you can install the Logic Apps Management Solution (Preview) in the Azure portal and set up Azure Monitor logs to collect diagnostic data. After you set up your logic app to send that data to an Azure Log Analytics workspace, telemetry flows to where the Logic Apps Management Solution can provide health visualizations. For more information, see [Set up Azure Monitor logs and collect diagnostics data for Azure Logic Apps](#). With diagnostics enabled, you can also use Azure Monitor to send alerts based on different signal types such as when a trigger or a run fails. For more information, see [Monitor run status, review trigger history, and set up alerts for Azure Logic Apps](#).
- For Standard logic app workflows, you can enable Application Insights support, which provides curated visualizations as a foundation for monitoring Azure services. These visualizations help you more effectively monitor Standard workflows by using dashboards specifically designed for Azure Logic Apps (Standard). The dashboard scope covers the workflows inside a Standard logic app. The dashboard is built on [Azure Workbooks](#) and offers various visualizations. You can easily extend and customize these workbooks to meet specific needs.

[Serverless 360](#) is an external solution from [Kovai](#) that provides monitoring and management through mapping Azure services, such as Azure Logic Apps, Azure Service Bus, Azure API Management, and Azure Functions. You can reprocess messages by using dead letter queues in Azure Service Bus, enable self-healing to address intermittent service disruptions, and set up proactive monitoring through synthetic transactions.

You can configure custom monitoring rules and view logs in a portal experience. You can send notifications through various channels, such as email, Microsoft Teams, and ServiceNow. To visually determine the health of your interfaces, service maps are available.

Business activity monitoring

The following section describes options to monitor and collect telemetry for workloads in BizTalk Server and Azure Integration Services.

BizTalk Server

BizTalk Server includes a feature called Business Activity Monitoring (BAM) that allows developers and business analysts to define tracking profiles that they can apply to orchestrations. As messages move through receive and send ports, data attributes are captured and stored in a BAM database. Custom implementation is also available through a .NET Fx API.

Azure Logic Apps

As a developer or business analyst working on solutions that integrate services and systems using various Azure resources, you might have difficulties visualizing the relationship between the technical components in your solution and your business scenario. To include business context about the Azure resources in your solution, you can build business processes that visually represent the business logic implemented by these resources. In [Azure Business Process Tracking](#), a business process is a series of stages that represent the tasks flowing through real-world business scenario.

Another option is that you can use an external solution from [Kovai](#) ↗ called [Serverless 360](#) ↗. Along with the monitoring platform, you can use the [business activity monitoring feature](#) ↗ that provides end-to-end tracking for business process flows across cloud-native and hybrid integrations. This feature includes a managed connector that developers can use to instrument code and capture important business data. Administrators can subsequently build dashboards and share them with business analysts.

Tracking

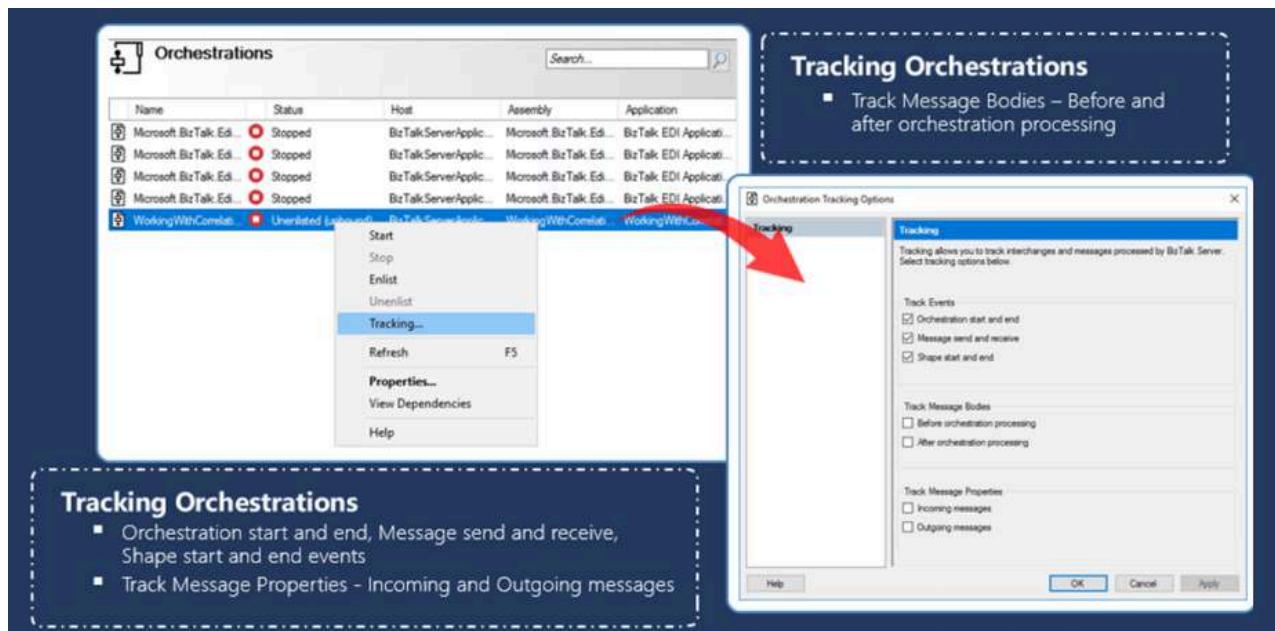
The following section describes options to track artifacts for performance monitoring and health analysis in BizTalk Server and Azure Integration Services.

BizTalk Server

- Message tracking

BizTalk Server administrators can use [message body tracking](#) to indicate when to persist message bodies to storage for troubleshooting and audit purposes. Message tracking is an expensive operation from both performance and storage perspectives, so use this capability selectively to avoid performance issues. When you enable message body tracking on Receive and Send ports, BizTalk Server copies the data to the [BizTalk Tracking](#)

database (BizTalkDTADb) by using the [SQL Server Agent job](#) named `TrackedMessages_Copy_<message-box-name>`.



You can apply tracking to almost all BizTalk Server artifacts, including orchestrations, pipelines, Receive ports, Send ports, schemas, and business rules. These options are enabled or disabled in runtime without affecting your code (solution) or requiring any restart.

- [Health and Activity Tracking \(HAT\)](#)

Although the HAT tool was removed from the BizTalk Server starting with the 2009 edition, the functionality still exists within the BizTalk Administration Console.

Administrators can search for data through the New Query interface within the Group Overview experience. You can tailor queries based on different criteria including the event type, port name, URI, schema name, and more. If you want to review the message bodies that moved through a Receive or Send port, you can access this information provided that you enabled port level tracking. For more information, see [Health and activity tracking](#).

- [Integration with Application Insights and Azure Event Hubs](#)

As of [BizTalk Server 2016 Feature Pack 1](#), you can publish telemetry to Application Insights in Azure Monitor or to Azure Event Hubs. This approach avoids SQL Server disk capacity issues, so that you can use elastic, cloud-based data stores instead, such as Application Insights, Log Analytics, and run history in Azure Logic Apps.

Azure Logic Apps

Azure Logic Apps also provides rich run history so that developers and support analysts can review action by action telemetry, including all processed inputs and outputs. To help protect any sensitive data, you can [enable secure inputs and outputs](#) on individual actions in workflows. This capability obfuscates or hides the data in logs and workflow run histories to avoid leaks.

Beyond data obfuscation, you can use [Azure RBAC](#) rules to protect data access. Azure RBAC includes [specific built-in roles for Azure Logic Apps \(Standard\)](#).

Beyond Azure RBAC, you can also [restrict access to run history in Azure Logic Apps by IP address range](#).

Hosting

The following section describes hosting options for BizTalk Server and Azure Integration Services.

BizTalk Server

BizTalk Server 2020 supports the following Microsoft platforms and products, starting with Cumulative Update 6:

- Windows Server 2022, Windows Server 2019, and Windows 11
- Visual Studio 2019 Enterprise and Visual Studio 2019 Professional
- SQL Server 2022, SQL Server 2019
- Office 2019 and Office 2016

You can install and run BizTalk Server on your own hardware, on-premises virtual machine, or Azure virtual machines. Azure virtual machines give you the flexibility of virtualization for a wide range of computing solutions with support for BizTalk Server, Windows Server, SQL Server, and more. All current generation virtual machines include load balancing and auto-scaling at no cost.

Azure Logic Apps

- Hosting plans

In single-tenant Azure Logic Apps, a Standard logic app is similar to an Azure function or web app where you can use a single Workflow Service plan to host multiple Standard logic apps. This similarity means you don't have to deploy all your workflows in a single Standard logic app resource. Instead, you can organize these workflows into logical groups (logic apps) to help you better manage other aspects of your solution. This

approach helps you get the most out of your Workflow Service plan and future-proof your applications, which you can implement so that they can individually scale.

A Standard logic app has the following pricing tiers: WS1, WS2 and WS3. Functionally, each tier provides the same capabilities. Your requirements for compute and memory determine is best for your scenario, for example:

[Expand table](#)

Pricing tier	Virtual CPU (vCPU)	Memory (GB)
WS1	1	3.5
WS2	2	7
WS3	4	14

For more information, see [Pricing tiers in the Standard model](#).

- Hybrid deployment model (preview)

Azure Logic Apps offers a hybrid deployment model so that you can deploy and host Standard logic app workflows in on-premises, private cloud, or public cloud scenarios. This model gives you the capabilities to host integration solutions in partially connected environments when you need to use local processing, data storage, and network access. With the hybrid option, you have the freedom and flexibility to choose the best environment for your workflows. For more information, see [Set up your own infrastructure for Standard logic apps using hybrid deployment \(Preview\)](#).

- Availability and redundancy

In Azure, [availability zones](#) provide resiliency, distributed availability, and active-active-active zone scalability. To increase availability for your logic app workloads, you can [enable availability zone support](#), but only when you create your logic app. You'll need at least three separate availability zones in any Azure region that supports and enables zone redundancy. The Azure Logic Apps platform distributes these zones and logic app workloads across these zones. This capability is a key requirement for enabling resilient architectures and providing high availability if datacenter failures happen in a region. For more information, see [Build solutions for high availability using availability zones](#).

- Isolated and dedicated environment

For Standard logic apps, you have the option to select an App Service Environment (ASE) v3 for your deployment environment. With an ASE v3, you get a fully isolated and dedicated environment to run applications at high scale with predictable pricing. You pay

only for the [ASE App Service plan](#), no matter how many logic apps that you create and run.

For scenarios that require additional Azure integration services, see the following documentation:

- [Azure Service Bus Premium and Standard messaging tiers](#)
- [Azure API Management - Feature-based tier comparison](#)
- [Azure Data Factory - Plan to manage costs](#) and [Understanding Data Factory pricing through examples](#)

Deployment

BizTalk Server

The native deployment packaging in BizTalk Server is based on a Microsoft Installer (MSI) file combined with an environment configuration, or bindings, file. These two files create a separation between the installation of components, which are deployed to the following BizTalk Server repositories and defines settings at the port and pipeline level, including endpoint, secrets, pipeline configuration, and others.

- Management DB
- BizTalk Server local folders
- .NET Global Assembly Cache

Although this process can prove effective, you also have to manage each individual environment configuration separately from the code. The BizTalk Deployment Framework (BTDF) open-source project offers one solution for this problem. With this tool, you can maintain the environment configuration as part of your BizTalk Server solution by using a tokenized binding file, which you create at design time, and a token matrix, which you create as an Excel file, for each environment.

The build process then creates a unified and versioned MSI file. This file supports component deployment and environment configuration from the same package, which gives you better control over the version of the solution that you want to implement across environments.

Support for a BTDF package in a continuous integration-continuous deployment (CI/CD) pipeline is available in BizTalk Server 2020, which includes this functionality introduced with the BizTalk Server 2016 Feature Packs. You can use this functionality and the Azure DevOps platform to streamline automatic deployment for BizTalk Server solutions across environments.

Azure Logic Apps

When you deploy an Azure Logic Apps resource or any other Azure Integration Services component or solution to Azure, you must manage the following items:

- Azure resources that act as containers or the infrastructure for the solutions that you want to deploy, for example, the Standard logic app resource, API Management instance, Service Bus namespace, or Event Grid topic
- The actual logic implemented by each component such as workflows, APIs, queues, and subscriptions
- The environment-specific configuration associated with each component, for example, permissions, secrets, alerts, and so on

When you keep the infrastructure definition separate from the code, you can treat the infrastructure definition as just another piece of code that you can version, store safely in a source control repository, and trigger a deployment when the definition changes. This practice, commonly known as Infrastructure as Code (IaC), improves environment quality because you can create versions for each environment and track changes back to source control.

Azure Logic Apps supports IaC by providing the capability to create infrastructure resources using Azure Resource Management templates. Although ARM templates can seem complex to understand and implement as a unified solution, you can use abstraction tools, such as Bicep, Terraform, or Pulumi, which provide a code-like experience for creating your infrastructure definition. Although these tools provide abstraction layers over ARM templates, the tools ultimately generate ARM templates and can deploy those templates for you.

With your infrastructure in place, you can deploy the logic that implements your end-to-end workflows. As Azure Integration Services offers a collection of tools for you to implement your integration workflows, you must deploy each component. For solutions built with Azure Integration Services, CI/CD pipelines are usually based on deploying an orchestration of components. DevOps engineers can use built-in actions that abstract deployment activities, or they use generic actions that run either CLI commands or automation scripts such as PowerShell and Bash. In most cases, engineers customize pipelines based on the application's needs, review guidance from official documentation, and use sample repositories as a starting point.

The process to get each component ready for deployment usually takes the following steps under consideration:

- Continuous integration phase
 1. Get the source code's latest version.
 2. Prepare the code with the environment-specific configuration.

The details for this step depend on each technology's support for external injection of environment variables. The basic premise is that environment-based configuration information, such as connection strings and references to external resources, are abstracted to reference an application settings repository. So, in this scenario, you'd store references that can exist as clear text directly in the application settings repository, but you'd store sensitive values, such as secrets, as reference pointers to entries in a secrets store, such as an Azure key vault.

Azure Logic Apps makes this approach possible for a Standard logic app resource by supporting references to the application settings repository, which you can then map name-value pairs to entries in your key vault.

3. Package the code for deployment in various environments.

- Continuous deployment phase

1. Deploy packaged code in the destination environment.
2. Update the application settings repository with the correct environment values, either as clear text or references to entries in your key vault.
3. Update any required permissions that depend on code.
4. Get your application ready for execution, if required.

Feature matchup

The following table and diagram roughly show how resources, artifacts, features, and capabilities compare and match up between BizTalk Server, Azure Logic Apps, and Azure Integration Services. While Azure Logic Apps is a key platform for integration workloads, make sure that you consider all the available capabilities in Azure Integration Services and in Azure as a whole.

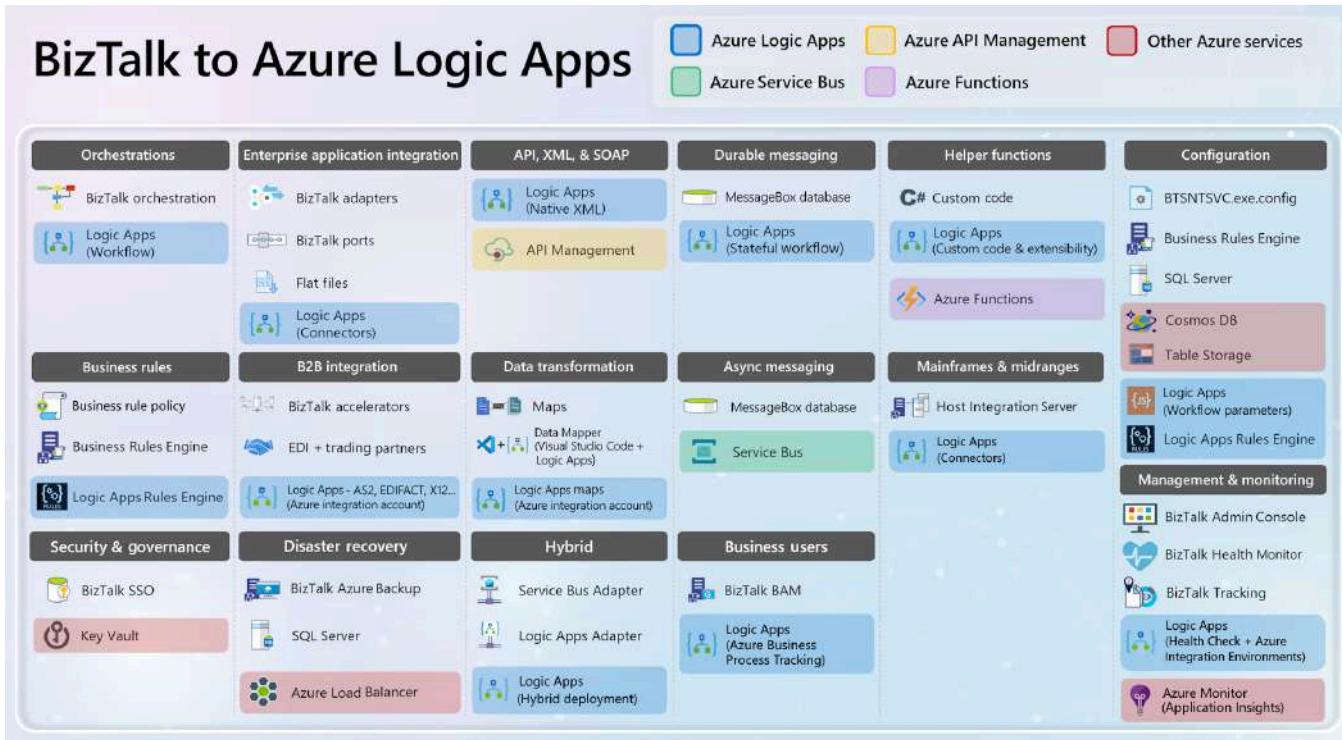
 [Expand table](#)

Feature or functionality	BizTalk Server	Azure
Orchestrations	<ul style="list-style-type: none">- BizTalk Server orchestration- C# code	<ul style="list-style-type: none">- Azure Logic Apps workflow- Azure Logic Apps workflow templates- Azure Functions function app
Pipelines	<ul style="list-style-type: none">- BizTalk Server pipelines- Pipeline components	<ul style="list-style-type: none">- Azure Logic Apps workflows (as pipelines)- Azure API Management (as pipelines)

Feature or functionality	BizTalk Server	Azure
		<ul style="list-style-type: none"> - Azure Functions function app - Azure API app
Message routing	<ul style="list-style-type: none"> - MessageBox - Property Promotions - Filters 	<ul style="list-style-type: none"> - Azure Service Bus queues and topics (message headers, message properties, and subscriptions) - Azure Event Grid or Azure API Management - SQL Server or Azure Cache for Redis
Application connectivity	<ul style="list-style-type: none"> - BizTalk Server out-of-the-box and custom adapters - Internet Information Services (IIS) and Azure API Management (hybrid capabilities) 	<ul style="list-style-type: none"> - Azure Logic Apps connectors - Azure API Management (as connectors) - Azure Functions function app - Azure API app
Cross-references	xref_* tables on BizTalk Management database (BizTalkMgmtDb)	<ul style="list-style-type: none"> - Azure Functions - SQL Server - Custom
Schemas (XSD)	<ul style="list-style-type: none"> - BizTalk Server schemas - XML, JSON, and flat file schemas 	<ul style="list-style-type: none"> - Azure Logic Apps (Standard) - Azure integration account - Azure storage account - Azure Functions function app - Azure API app
Maps	<ul style="list-style-type: none"> - BizTalk Mapper - XSLT maps - Azure API Management (hybrid capabilities) 	<ul style="list-style-type: none"> - Azure Logic Apps (Standard) - XSLT maps, Liquid templates - Azure integration account (XSLT maps, Liquid templates) - Azure storage account - Azure Functions function app - Azure API app - Data Mapper tool (Azure Logic Apps Standard extension for Visual Studio Code)
Business rules	BizTalk Server Business Rules Engine	Azure Logic Apps Rules Engine
Business activity monitoring	BizTalk Server Business Activity Monitoring	Azure Business Process Tracking
EDI	<ul style="list-style-type: none"> - BizTalk Server out-of-the-box capabilities - Parties, partners, agreements, AS2, X12, EDIFACT 	Azure Logic Apps and Azure integration account (partners, agreements, AS2, X12, EDIFACT)
HL7, RosettaNet, and SWIFT	BizTalk Server accelerators for HL7, RosettaNet, and SWIFT	- Azure Logic Apps, Azure integration account, RosettaNet and SWIFT connectors

Feature or functionality	BizTalk Server	Azure
		<ul style="list-style-type: none"> - Azure API Management for FHIR (HL7) - Azure Blueprint, which enables SWIFT CSP compliance on Azure
Secrets	Enterprise Single Sign-On (SSO)	<ul style="list-style-type: none"> - Azure Key Vault - SQL Server - Application configuration
Security and governance	<ul style="list-style-type: none"> - Enterprise Single Sign-On (SSO) - SSO affiliate applications - Active Directory - Signing certificates - IIS Security Authentication - Network security 	<ul style="list-style-type: none"> - Microsoft Entra ID - Azure Network Security - Azure role-based access control (Azure RBAC) - Claims, tokens - Shared Access Policies
Data configuration	<ul style="list-style-type: none"> - Config files - Enterprise SSO application configuration - Custom cache components - Custom database - Business Rules Engine - Windows registry 	<ul style="list-style-type: none"> - Azure Key Vault - Azure App Configuration - Azure Cosmos DB - Azure Table Storage - Azure Logic Apps (Standard) configuration - Azure Functions configuration - Azure API Management named values and backends - SQL Server - Custom caching - Custom database
Deployment	<ul style="list-style-type: none"> - BizTalk Server binding file 	<ul style="list-style-type: none"> - Azure Pipelines - Bicep scripts - Terraform
Tracking	<ul style="list-style-type: none"> - BizTalk Server tracking capabilities (Receive ports, Send ports, pipelines, orchestrations) - IIS tracking - Azure API Management built-in analytics (hybrid capabilities) 	<ul style="list-style-type: none"> - Azure Logic Apps run history and tracked properties - Azure Storage Account - Azure Monitor (Application Insights) - Azure API Management built-in analytics - Custom solution, for example, Azure Event Hubs plus Azure Functions plus SQL Server plus Azure Data Explorer
Monitoring	<ul style="list-style-type: none"> - BizTalk Administration Console - BizTalk Health Monitor 	Azure Monitor (Application Insights, Log Analytics)
Operations	<ul style="list-style-type: none"> - BizTalk Server Administration Console - Azure Pipelines 	<ul style="list-style-type: none"> - Azure portal - Azure Monitor - Azure Resource Manager templates

Feature or functionality	BizTalk Server	Azure
	<ul style="list-style-type: none"> - MSI, PowerShell - BizTalk Deployment Framework 	<ul style="list-style-type: none"> - Azure Pipelines - PowerShell, CLI, Bicep



To stay updated about the latest investments, subscribe to the [Integrations on Azure Blog - Tech Community](#).

Next steps

You've learned more about how Azure Logic Apps compares to BizTalk Server. Next, learn how to choose the best Azure capabilities for your scenarios. Or, skip ahead to review suggested approaches and resources, planning considerations, and best practices for your migration.

[Choose the best Azure Integration Services offerings for your scenario](#)

[Migration approaches for BizTalk Server to Azure Logic Apps](#)

Choose the best integration services in Azure for enterprise integration scenarios

07/05/2025

Azure Integration Services offers many capabilities across this collection of integration services, but some overlapping capabilities might exist. This guide provides information to help you choose the best services for your enterprise integration scenarios and requirements. Remember also to consider the full impact of using a particular service, including performance requirements, skill set availability, operational support, and costs.

(!) Note

If you're a BizTalk Server customer looking to move your workloads to Azure Logic Apps, you can get a migration overview and compare the capabilities between these two offerings by reviewing [Why migrate from BizTalk Server to Azure Logic Apps?](#)

When to choose a specific integration service and why

 [Expand table](#)

Service	When to choose	Why
Azure Logic Apps	You have business processes to orchestrate across multiple systems that span from legacy systems to artificial intelligence workloads. You need to migrate from Microsoft BizTalk Server or other integration platforms.	<ul style="list-style-type: none">- Provides greater developer productivity through the low-code workflow designer.- Excels at wiring API calls together quickly using prebuilt, out-of-the-box connectors.- Supports both synchronous and asynchronous processing.- Offers rich debugging history for stateful workflows.- Supports stateless workflows for low latency requirements.- Supports creating custom APIs and custom connectors, which let you wrap existing REST APIs or SOAP APIs to access services where

Service	When to choose	Why
		<p>no prebuilt connector currently exists. (Consumption workflows only)</p> <ul style="list-style-type: none"> - Supports creating custom built-in connectors based on a service provider. (Standard workflows only)
Azure Functions	<p>You need to build central utility functions that you can access from other integration platform components, such as Azure Logic Apps.</p> <p>You have unique data transformation requirements.</p>	<p>Provides an event driven, compute-on-demand experience for developers that need to extend the Azure application platform by implementing code triggered by events in Azure or other services and on-premises systems.</p>
Azure Data Factory	<p>You need the capability to transform and move large datasets across various data sources, such as file systems, database, SAP, Azure Blob Storage, Azure Data Explorer, Oracle, DB2, Amazon RDS, and more.</p>	<ul style="list-style-type: none"> - Provides a cloud-based serverless ETL service for scale-out, dataset integration, and data transformation. Can handle large data and message processing requirements. - Offers code-free UI for intuitive authoring and single-pane-of-glass monitoring and management. - Supports lift-and-shift for existing SQL Server Integration Services (SSIS) packages to Azure and running them with full compatibility in Azure Data Factory. The SSIS Integration Runtime offers a fully managed service, so you don't have to worry about infrastructure management.
Azure Service Bus	<p>You need a messaging system that supports the publish-subscribe model, ordered delivery, duplicate detection, message scheduling, and message expiration scenarios.</p>	<ul style="list-style-type: none"> - Provides a fully managed enterprise message broker with message queues and publish-subscribe topics. - By decoupling applications and services from each other, this service provides the following benefits: <ul style="list-style-type: none"> --- Load balancing across competing workers --- Safe message routing, data transfer, and control across service and application boundaries --- Coordinated transactional work that requires a high degree of reliability. - Complements Azure Logic Apps and

Service	When to choose	Why
		supports scenarios where you want to use SDKs, not connectors, to interact with Service Bus entities.
Azure Event Grid	You need an event subscription architecture to stay updated on state changes in one or more applications and systems because your integration solutions depend heavily on events to communicate such changes and make any related data changes.	<ul style="list-style-type: none"> - Provides a highly scalable, serverless event broker for integrating applications using events. Event Grid delivers events to subscriber destinations such as applications, Azure services, or any endpoint where Event Grid has network access. Event sources can include applications, SaaS services, and Azure services. - Increases efficiency by avoiding constant polling to determine state changes. As more underlying services emit events, subscription architecture increases in popularity.
Azure API Management	You want to abstract and protect your underlying service implementation in Azure Logic Apps from end users and consumers.	<ul style="list-style-type: none"> - Provides a hybrid, multi-cloud management platform for APIs across all environments. - Offers the capability to reuse central services in a secure way, giving your organization more governance and control over who can call enterprise services and how to call them. You can subsequently call these APIs from Azure Logic Apps after your organization catalogs them in Azure API Management.

Next steps

You've now learned more about which offerings in Azure Integration Services best suit specific scenarios and needs. If you're considering moving from BizTalk Server to Azure Integration Services, learn more about migration approaches, planning considerations, and best practices to help with your migration project.

[Migration approaches for BizTalk Server to Azure Logic Apps](#)

Migration approaches for BizTalk Server to Azure Logic Apps

06/03/2025

This guide covers migration strategies and resources along with planning considerations and best practices to help you deliver successful migration solutions.

⚠ Note

For a migration overview and guide to choosing services in Azure for your migration, review the following documentation:

- [Why migrate from BizTalk Server to Azure Logic Apps?](#)
- [Choose the best integration service in Azure for your scenario](#)

Strategy options

The following sections describe various migration strategies along with their benefits and disadvantages:

Big bang

A "big bang" or "direct changeover" is an approach that requires lots of planning and isn't recommended for organizations that are unfamiliar with Azure Logic Apps or that have large systems or solutions to migrate. When an organization implements a new technology stack, new learnings usually often result. By investing too early or too much, you won't have the opportunity to benefit from lessons learned and adjust without risking significant rework.

This approach might also take longer to reap or accrue value. If you've already completed some migration activities, but you haven't yet released them into production due to other pending or in-progress work, your migrated artifacts aren't generating value for your organization.

We recommend that you consider this approach only if you have small, low complexity BizTalk workloads, subject matter experts (SMEs) who know your BizTalk environment, and direct mappings between the BizTalk features that you currently use and Azure Logic Apps. Experience with Azure Logic Apps also considerably reduces the risks with following this approach.

Iterative or wave based (Recommended)

This approach provides the opportunity for your organization to incrementally achieve value, but sooner than they might otherwise. Your project team can learn about the technology stack early by using retrospectives. For example, you can deploy an existing BizTalk interface or project to production and then learn about the solution's needs, which include management, scalability, operations, and monitoring. After you gain this knowledge, you can plan sprints to optimize existing capabilities or introduce new patterns that you can subsequently use in future work.

Regardless of your approach, if you plan on moving to Azure Logic Apps or Azure in general, strongly consider refactoring your BizTalk Server solutions into serverless or cloud-native solutions before you decommission your server infrastructure. This choice is an excellent strategy if your organization wants to transform the business completely to the cloud.

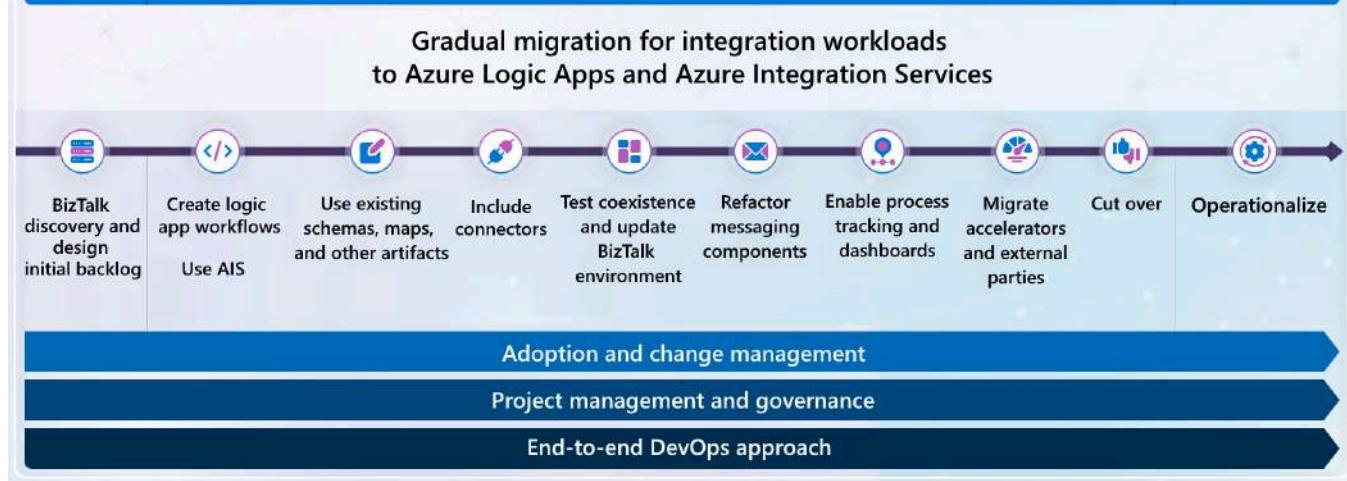
BizTalk Server and Azure Logic Apps have different architectures. To further modernize your solutions, you can use Azure Integration Services to extend the capabilities in Azure Logic Apps that address core customer integration needs.

For a higher return on investment (ROI), we recommend that any BizTalk migration use the core native capabilities in Azure Logic Apps (Standard) as much as possible and extended with other Azure Integration Services as needed. This combination makes additional scenarios possible, for example:

- Cloud native hybrid capabilities with Azure Logic Apps (Standard) with hybrid deployment
- Stateful or stateless workflow capabilities in Azure Logic Apps (Standard)
- Native, built-in (in-app) mainframe and midranges integration with connectors in Azure Logic Apps (Standard)
- Pub-sub messaging using Azure Service Bus
- Advanced SOAP capabilities in Azure API Management

Deliver a BizTalk migration project

To complete such a project, we recommend that you follow the iterative or wave-based approach and use the [Scrum process](#). While Scrum doesn't include a Sprint Zero (Sprint 0) concept for pre-sprint activities, we recommend that you focus your first sprint on team alignment and technical discovery. After Sprint 0, follow the execution of multiple migration sprints and focus on releasing features towards a minimum viable product (MVP).



Sprint 0

During this sprint, we recommend that you execute BizTalk Server Environments Discovery with Waves Planning. Understanding the project's breadth and depth is critical for success. The following list includes the specific areas to address during Sprint 0:

[Expand table](#)

Area	Description
Discovery	<p>Capture data about all your interfaces and applications so you can learn the number of interfaces and applications that you need to migrate. You also need to assign complexity to each interface or process. During this cataloging process, collect the following information to prioritize work:</p> <ul style="list-style-type: none"> - Adapters in use - BizTalk Server features in use, such as Business Activity Monitoring, Business Rules Engine, EDI, and so on - Custom code, such as expressions, maps, and pipeline components - Message throughput - Message sizes - Dependencies - Application and system dependencies
Architecture design	Create the high-level architecture to use as the focal point for the migration. This design includes elements that address high-level functional and non-functional needs.

Area	Description
Minimum viable product (MVP) definition	Define the first wave features. In other words, the processes that need support after you complete the first wave.
Initial migration backlog	Define the first wave features and their work items with technical elaboration.

Discovery tools

To help you with migration discovery, you can use the Azure Integration Migrator command-line tool, also called the [BizTalk Migration tool](#), which is a Microsoft open-source project. This tool uses a phased approach to help you uncover useful insights and strategies for migrating your solutions to the cloud. We recommend using the migrator tool only for discovery and report generation. You might also want to consider using other products for discovery from partners who provide solutions in this space.

For another way to generate an inventory with BizTalk Server elements, you can use [The BizTalk Documenter](#), which is developed by Mark Brimble. This tool works with BizTalk Server 2020, despite stating that only BizTalk Server 2016 is supported.

Architecture design

While Azure Logic Apps provides capabilities that let you reuse BizTalk Server assets, you must have a modern architecture design to embrace the benefits from more modern capabilities. From a functional perspective, use your business logic as much as possible. From a product modernization perspective, use Azure Integration Services as much as you can. For quality and cross-cutting concerns, we recommend that you use the [Azure Well-Architected Framework](#).

Under this framework, BizTalk migrations are [mission-critical workloads](#). This term describes collections of application resources that require high availability on the platform, meaning that they must always be available, operational, and resilient to failures.

To complete the architecture design for your BizTalk migration, follow [Design methodology for mission-critical workloads on Azure](#). For an initial architecture and topology, review and use the reference architecture described in [Basic enterprise integration on Azure](#) in the [Azure Architecture Center](#).

To set up your initial environment, use the [Azure Integration Services Landing Zone Accelerator](#), which is targeted for building and deploying an integration platform using a typical enterprise landing zone design.

Minimum viable project (MVP) definition

An MVP is a product version that has just enough customer-usable features. This version shows the product's possibilities and potential to gather customer feedback and continue the work. For a BizTalk migration, your MVP definition reflects the iterations, waves, or groups of sprints that you need to make progress towards working features and to meet initial integration workloads.

We recommend that your MVP definition include the following business outcomes, which are expressed as *Epics* in Scrum terminology:

Business outcomes (Epics)

- What is the primary goal that you want to achieve?
- What capabilities or features must you address for this MVP?
- What are the business process flows? This question provides the opportunity to optimize existing processes supported by BizTalk Server.
- What are the business decisions, for example, business outcomes that affect the MVP, or what is the resource availability?

We recommend that your MVP include the following in-scope processes, which are expressed as *Features* in Scrum terminology:

In-scope processes (Features)

Expand table

Feature	Description
High-level system functionality	You can extract this information using the discovery tools and express the descriptions in terms of features.
Actors or personas	Use this information to determine the individuals affected by the MVP's supported scenarios.
Orchestrations	You can extract this information using the discovery tools.
Data entities and messages	These elements give you an opportunity to learn whether you can include further improvements in the data exchanged by the BizTalk Server environment.
Data mappings	Today's world relies on JSON. However, BizTalk Server uses XML. This moment is a great opportunity to decide the data format and conversion needs for the new platform.
Business rules	These data-centric rules open an opportunity for you to rethink their approach or reuse them by employing Azure Logic Apps capabilities.

Feature	Description
Data privacy considerations	You must make privacy a top priority. Unless your customer chooses the hybrid deployment model in Azure Logic Apps (Standard), you must address this area in each wave due to potential deployment environment changes.
Regulatory considerations	This aspect is more relevant if your customers don't have cloud-based workloads.
Secure by design	You must design each feature with security in mind.
Proposed features for coexistence	When you deliver each wave, you have a certain degree of coexistence. You must align this hybrid architecture with existing Service Level Indicators (SLIs) and Service Level Objectives (SLOs).
Non-functional considerations	Business processes might have different non-functional requirements. Not everything must happen in real time. Conversely, not everything is a batch process.
Business metrics	An optional opportunity to show progress for the migration work.

You'll also want to identify and list the various out-of-scope variables that shape the scope of your MVP work, for example:

- Resource availability
- Risks
- Documentation
- Time to market

Initial backlog

The [*initial backlog*](#) is a set of User Stories, which you group into Features to build the [*in-scope processes*](#) for your MVP. In other words, an MVP is represented by Scrum items known as Epics, Features, and User Stories. Ideally, each Epic encompasses a group of BizTalk applications or BizTalk projects. You can use the simple rule that associates one BizTalk application or BizTalk project with a feature.

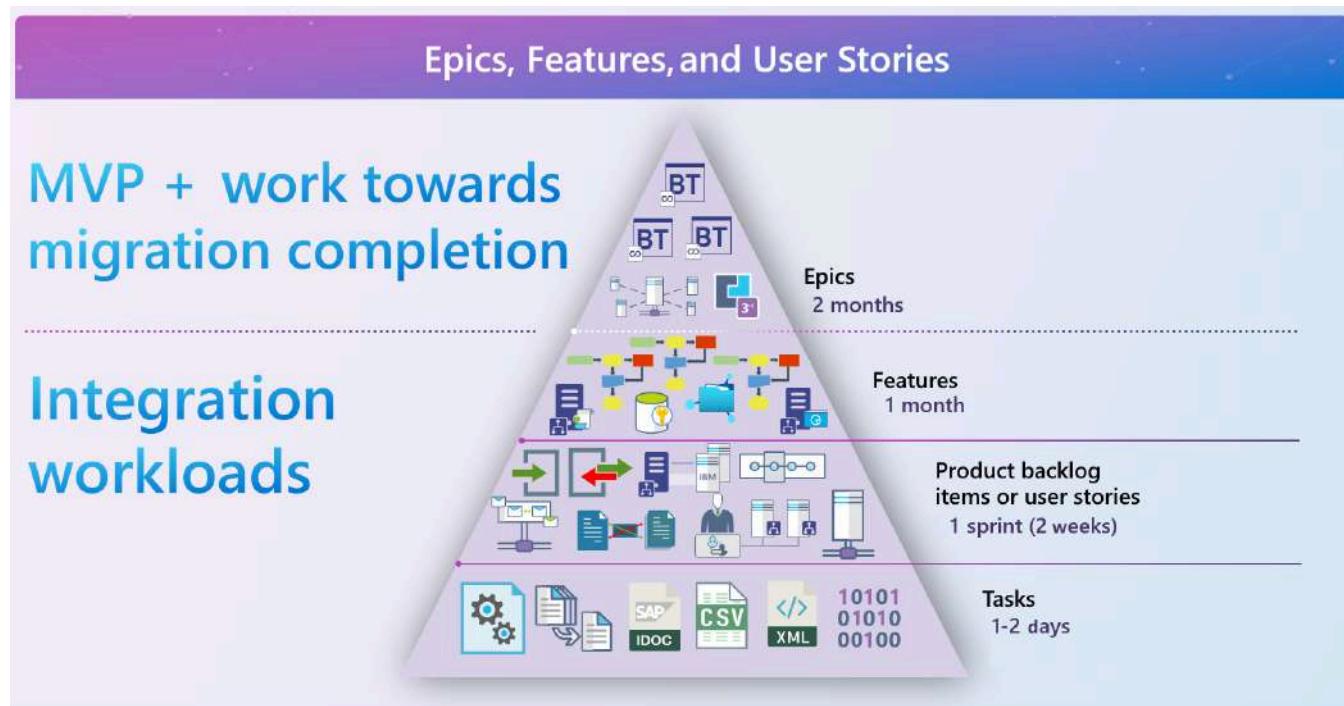
For example, suppose you have a BizTalk Server project with an orchestration called "LoanRequests" that customers use to request bank loans. So, you have the following proposed Feature and User Story:

- **Feature:** Loan processing
- **User Story:** "As a customer, I want to submit a loan application so the bank can add funds to my secure account."

This User Story, which might currently exist as an implementation in BizTalk Server, has the following tasks to implement using Azure Logic Apps and Azure Integration Services:

- Collect BizTalk reusable artifacts.
- Create a loan request workflow using Azure Logic Apps.
- Configure asynchronous messaging using Azure Service Bus or IBM MQ.
- Map JSON to XML data using an Azure Logic Apps workflow.
- Customize Azure Integration Services as required for messaging patterns.

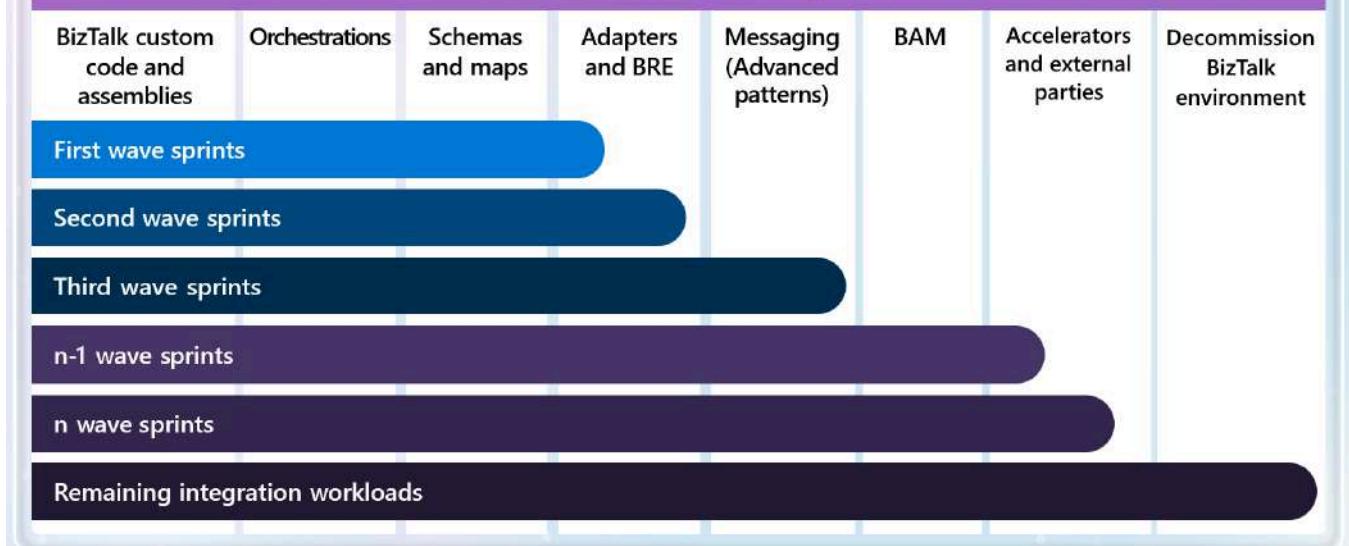
The following diagram shows the suggested durations for Epics, Features, User Stories, and Tasks, which subdivide User Stories. Although implementation decisions affect these durations, they assume that you are using existing BizTalk artifacts in Azure Logic Apps. Create your Standard workflows by using the [prebuilt workflow templates](#) as much as possible.



Migration waves (Sprints)

After your team completes Sprint 0, you should have a clear view of the MVP to build. A *wave* is a set of sprints. Your initial backlog should include work items that follow the next diagram as much as possible:

Each wave coexists with the existing BizTalk environment

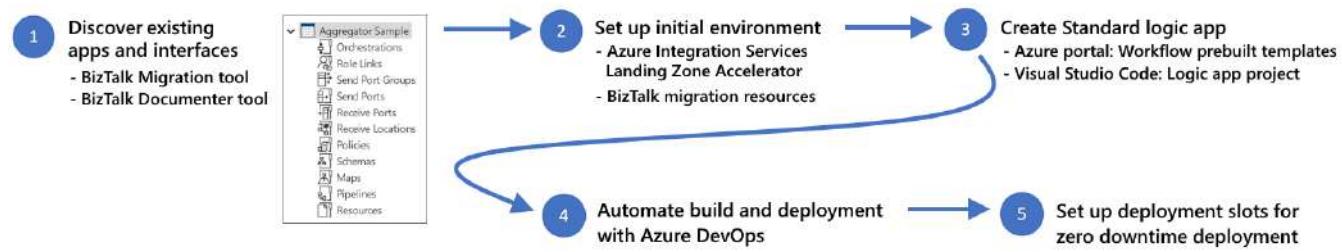


During a wave, your team completes the activities to migrate, test, and release to production. Let's more closely examine what happens in each wave.

Migrate

During each wave, migration focuses on the agreed User Stories. For the first wave, your team focuses on the initial backlog. Technology decisions must use the information in the BizTalk Server features mapping, described by [Feature matchup - Why migrate from BizTalk Server to Azure Logic Apps?](#)

The following diagram shows the events that should happen during migration waves:



[Expand table](#)

Step Description

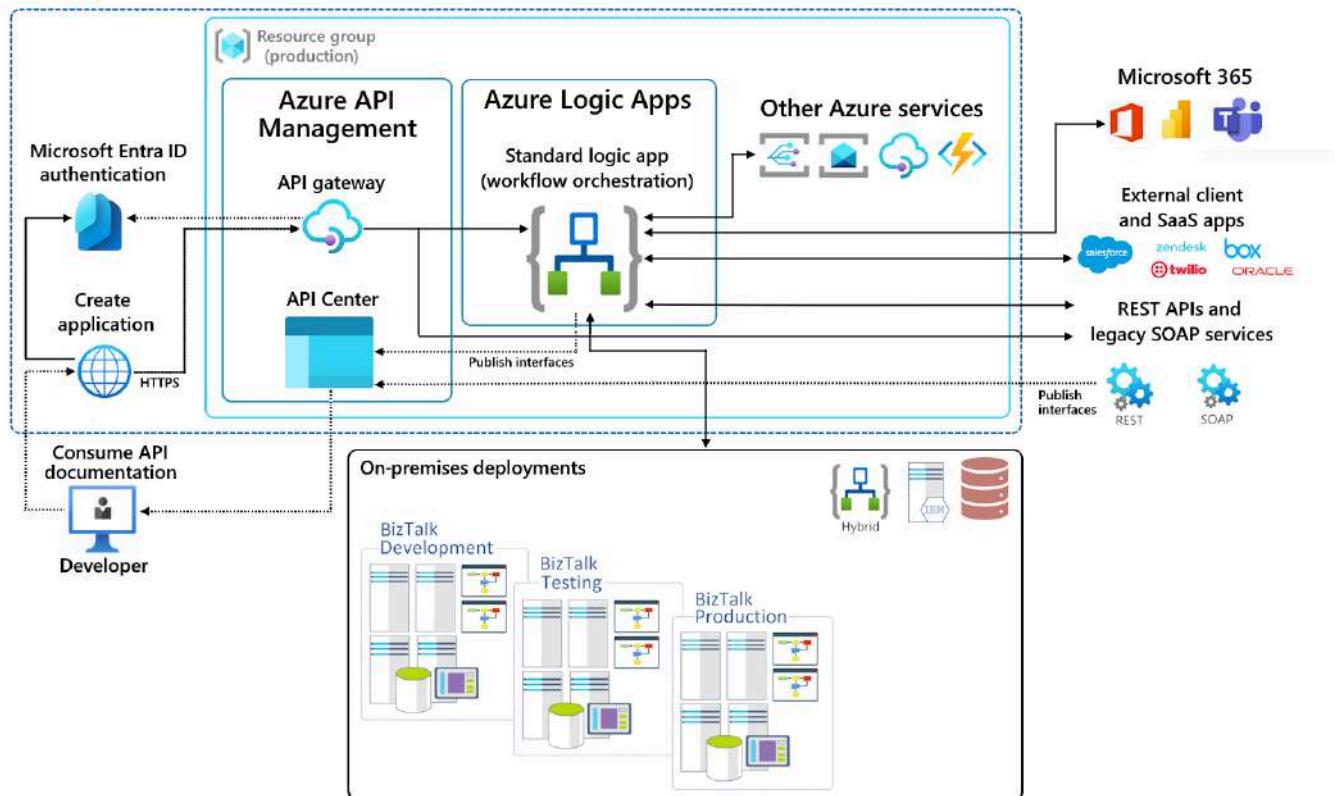
- 1 Discover existing BizTalk apps and interfaces. Although introduced in Sprint 0, this activity should happen when each wave starts. Customers might continue making changes in your BizTalk environment.

Resources:

- [BizTalk Migration tool](#)
- [BizTalk Documenter tool](#)

Step	Description
2	<p>Set up your initial migration environment. You can use the Azure Integration Services Landing Zone Accelerator, which is a cloud adoption framework for building and deploying an integration platform that has a typical enterprise landing zone design. As the workload owner, you can confidently achieve your target technical state by using the provided architectural guidance and BizTalk migration resources.</p> <p>For an example architecture, see Example migration environment.</p>
3	<p>Create and test Standard logic app workflows that run in single-tenant Azure Logic Apps using either the Azure portal or Visual Studio Code with the Azure Logic Apps (Standard) extension. With Visual Studio Code, you can locally develop, test, and store your logic app project using any source control system.</p> <p>For more information, see the following documentation:</p> <ul style="list-style-type: none"> - Create an example Standard logic app workflow using the Azure portal - Create an example Standard logic app workflow using Visual Studio Code
4	<p>For a diagram that shows an example logic app and connections, see Example migration environment.</p> <p>To get the full benefits from easily and consistently deploying your Standard logic app workflows across different environments and platforms, you must also automate your build and deployment process. The Azure Logic Apps (Standard) extension for Visual Studio Code provides tools for you to create and maintain automated build and deployment processes using Azure DevOps.</p> <p>For more information, see Automate build and deployment for Standard logic app workflows with Azure DevOps.</p>
5	<p>To deploy mission-critical Standard logic apps that are always available and responsive, even during updates or maintenance, enable zero downtime deployment by creating and using deployment slots. Zero downtime means that when you deploy new versions of your app, end users shouldn't experience disruption or downtime.</p> <p>For more information, see Set up deployment slots to enable zero downtime deployment in Azure Logic Apps.</p>

The following diagram shows an example initial migration environment with a Standard logic app that orchestrates workflows that communicate with APIs, services, hybrid solutions, and on-premises resources:



Test

Each *wave* has its own testing activities, which are embedded in each User Story. If you want to use [shift-left testing](#), make sure that you complete the following tasks:

- Automate your tests.

Azure Logic Apps (Standard) includes the capability to perform automated testing. The following list includes more information and resources that are freely available on GitHub:

- [Automated testing with Azure Logic Apps \(Standard\)](#) ↗ from the Azure Logic Apps team

With Azure Logic Apps (Standard), automated testing is no longer difficult to perform, due to the underlying architecture, which is based on the Azure Functions runtime and can run anywhere that Azure Functions can run. You can write tests for workflows that run locally or in a CI/CD pipeline. For more information, see the sample project for the [Azure Logic Apps Test Framework](#) ↗.

This test framework includes the following capabilities:

- Write automated tests for end-to-end functionality in Azure Logic Apps.
- Perform fine-grained validation at the workflow run and action levels.
- Check tests into a Git repo and run either locally or within CI/CD pipelines.
- Mock testing capabilities for HTTP actions and Azure connectors.

- Configure tests to use different setting values from production.
- [Integration Playbook: Logic Apps Standard Testing](#) from Michael Stephenson, Microsoft MVP

The [Integration Playbook testing framework](#) builds on the Microsoft-provided test framework and supports additional scenarios:

- Connect to a workflow in a Standard logic app.
- Get the callback URL so that you can trigger the workflow from a test.
- Check the results from the workflow run.
- Check the operation inputs and outputs from the workflow's run history.
- Plug into automated testing frameworks that logic app developers might use.
- Plug into SpecFlow to support behavior-driven-development (BDD) for logic apps.

Regardless which automation approaches or resources that you use, you're well on your way to having repeatable, consistent, and automated integration tests.

- Set up mock response testing using static results.

Regardless whether you set up automated tests, you can use the [static results capability](#) in Azure Logic Apps to temporarily set mock responses at the action level. This functionality lets you emulate the behavior from a specific system that you want to call. You can then perform some initial testing in isolation and reduce the amount of data that you'd create in line of business systems.

- Run side by side tests.

Ideally, you already have baseline integration tests for your BizTalk Server environment and established automated tests for Azure Logic Apps. You can then run tests side-by-side in a way that helps you check your interfaces by using the same data sets and improve overall test accuracy.

Release to production

After your team finishes and meets the "definition of done" for the User Stories, consider the following tasks:

1. Create a communication plan for your release to production.
2. Make a "cut-over" plan.

A cut-over plan covers the details about the tasks and activities necessary to switch from the current platform to the new platform, including the steps that your team plans to execute. Include the following considerations in your cut-over plan:

- Prerequisite steps
- Dress rehearsal
- People
- Schedule estimates
- Disabling interfaces in the old platform
- Enabling interfaces in the new platform
- Validation testing

3. Determine a rollback plan.

4. Run validation testing.

5. Plan for operations or production support.

6. Choose "go or no go" criteria for releasing to production.

7. Celebrate your team's success.

8. Hold a retrospective.

Best practices for a BizTalk migration

While best practices might vary across organizations, consider a conscious effort to promote consistency, which helps reduce unnecessary efforts that "reinvent the wheel" and the redundancy of similar common components. When you help enable reusability, your organization can more quickly build interfaces that become easier to support. Time to market is a key enabler for digital transformation, so a top priority is reducing unnecessary friction for developers and support teams.

When you establish your own best practices, consider aligning with the following guidance:

General naming conventions for Azure resources

Make sure to set up and consistently apply good naming conventions across all Azure resources from resource groups to each resource type. To lay a solid foundation for discoverability and supportability, a good naming convention communicates purpose. The most important point for naming conventions is that you have them, and that your organization understands them. Every organization has nuances that they might have to take into account.

For guidance around this practice, review the following Microsoft recommendations and resources:

- [Abbreviation examples for Azure resources](#)

- [Azure Naming Tool](#), which generates Azure-compliant names, helps you standardize on names, and automates your naming process.

Naming conventions for Azure Logic Apps resources

The design for your logic app and workflow provides a key starting point because this area provides flexibility for developers to create unique names.

Logic app resource names

To differentiate between Consumption and Standard logic app resources, you can use different abbreviations, for example:

- Consumption: **LACon**
- Standard: **LAStd**

From an organizational perspective, you might design a naming pattern that includes the business unit, department, application, and optionally, the deployment environment, such as **DEV**, **UAT**, **PROD**, and so on, for example:

```
LAStd-<*business-unit-name*>-<*department-name*> or *application-name*>-<*environment-name*>
```

Suppose you have a Standard logic app in development that implements workflows for the HR department in the Corporate Services business unit. You might name the logic app resource **LAStd-CorporateServices-HR-DEV**, and use [Pascal Case notation](#) where appropriate for consistency.

Logic app workflow names

A Consumption logic app resource always maps to only one workflow, so you only need a single name. A Standard logic app resource can include multiple workflows, so design a naming convention that you can also apply to member workflows. For these workflows, consider a naming convention based on the process name, for example:

```
Process-<*process-name*>
```

So, if you had a workflow that implements employee onboarding tasks, such as creating an employee record, you might name the workflow **Process-EmployeeOnboarding**.

Here are more considerations for designing your workflow naming convention:

- Follow the parent-child pattern for workflows where you want to highlight some relationship between one or more workflows.
- Take into account whether a workflow publishes or consumes a message.

Workflow operation names

When you add a trigger or action to your workflow, the designer automatically assigns the default generic name for that operation. However, operation names must be unique within your workflow, so the designer appends sequential numerical suffixes on subsequent operation instances, which makes readability and deciphering the developer's original intent difficult.

To make operation names more meaningful and easier to understand, you can add a brief task descriptor after the default text and use Pascal Case notation for consistency. For example, for the Parse JSON action, you can use a name such as **Parse JSON-ChangeEmployeeRecord**. With this approach or other similar approaches, you'll continue to remember that the action is **Parse JSON** and the action's specific purpose. So, if you need to use this action's outputs later in downstream workflow actions, you can more easily identify and find those outputs.

Note

For organizations that extensively use expressions, consider a naming convention that doesn't promote using whitespace (' '). The expression language in Azure Logic Apps replaces whitespace with underscores ('_'), which might complicate authoring. By avoiding spaces upfront, you help reduce friction when authoring expressions. Instead, use a dash or hyphen ('-'), which provides readability and doesn't affect expression authoring.

To avoid later possible rework and problems around downstream dependencies, which are created when you use operation outputs, rename your operations immediately when you add them to your workflow. Usually, downstream actions are automatically updated when you rename an operation. However, Azure Logic Apps doesn't automatically rename custom expressions that you created before you perform the rename.

Connection names

When you create a connection in your workflow, the underlying connection resource automatically gets a generic name, such as **sql** or **office365**. Like operation names, connection names must also be unique. Subsequent connections with the same type get a sequential numerical suffix, for example, **sql-1**, **sql-2**, and so on. Such names don't provide any context, which makes differentiating and mapping connections to their workflows extremely challenging, especially for developers who don't know the solution space and have to maintain these workflows.

So, meaningful and consistent connection names are important for the following reasons:

- Readability
- Easier knowledge transfer and supportability
- Governance

Again, having a naming convention is critical, although the format isn't overly important. For example, you can use the following pattern as a guideline:

`CN-<*connector-name*>-<*logic-app-or-workflow-name*>`

As a concrete example, you might rename a Service Bus connection in an **OrderQueue** logic app workflow with **CN-ServiceBus-OrderQueue** as the new name. For more information, see the Turbo360 (Formerly Serverless360) blog post, [Logic app best practices, tips, and tricks: #11 connectors naming convention](#).

Handle exceptions with scopes and "Run after" options

Scopes provide the capability to group multiple actions so that you can implement Try-Catch-Finally behavior. On the designer, you can collapse and expand a scope's contents to improve developer productivity.

When you implement this pattern, you can also specify when to run the **Scope** action and the actions inside, based on the preceding action's execution status, which can be **Succeeded**, **Failed**, **Skipped**, or **TimedOut**. To set up this behavior, use the **Scope** action's [Run after \(runAfter\) options](#):

- Is successful
- Has failed
- Is skipped
- Has timed out

Consolidate shared services

When you build integration solutions, consider creating and using shared services for common tasks. You can have your team build and expose a collection of shared services that your project team and others can use. Everyone gains increased productivity, uniformity, and the capability to enforce governance on your organization's solutions. The following sections describe some areas where you might consider introducing shared services:

Shared service	Reasons
Centralized logging	Provide common patterns for how developers instrument their code with appropriate logging. You can then set up diagnostic views that help you determine interface health and supportability.
Business tracking and business activity monitoring	Capture and expose data so that business subject matter experts can better understand the state of their business transactions and perform self-service analytic queries.
Configuration data	Separate your application configuration data from your code so that you can more easily move your application between environments. Make sure to provide a unified consistent and easily replicable approach to access configuration data so that project teams can focus on solving the business problem rather than spending time on application configurations for deployment. Otherwise, if every project approached this separation in a unique way, you can't benefit from economies of scale.
Custom connectors	Create custom connectors for internal systems that don't have prebuilt connectors in Azure Logic Apps to simplify for your project team and others.
Common datasets or data feeds	Expose common datasets and feeds as APIs or connectors for project teams to use, and avoid reinventing the wheel. Every organization has common data sets that they need to integrate systems in an enterprise environment.

Next steps

You've now learned more about available migration approaches and best practices for moving BizTalk Server workloads to Azure Logic Apps. To provide detailed feedback about this guide, you can use the following form:

[Give feedback about migration guidance for BizTalk Server to Azure Logic Apps](#)

Migrate workloads for Azure VMware Solution

Article • 12/01/2022

Azure VMware Solution lets you seamlessly migrate VMware workloads from your data center to Azure and integrate more Azure services with ease. You can manage your IT environments with the same VMware solution tools you already know at the same time. You have the choice and flexibility to determine what workloads to migrate, and you decide the right time to migrate them. With platform symmetry, you have complete control to transform based on how your organization defines its unique cloud journey.

Migrating VMware workloads to Azure can accelerate the standard methodology outlined in the Cloud Adoption Framework.

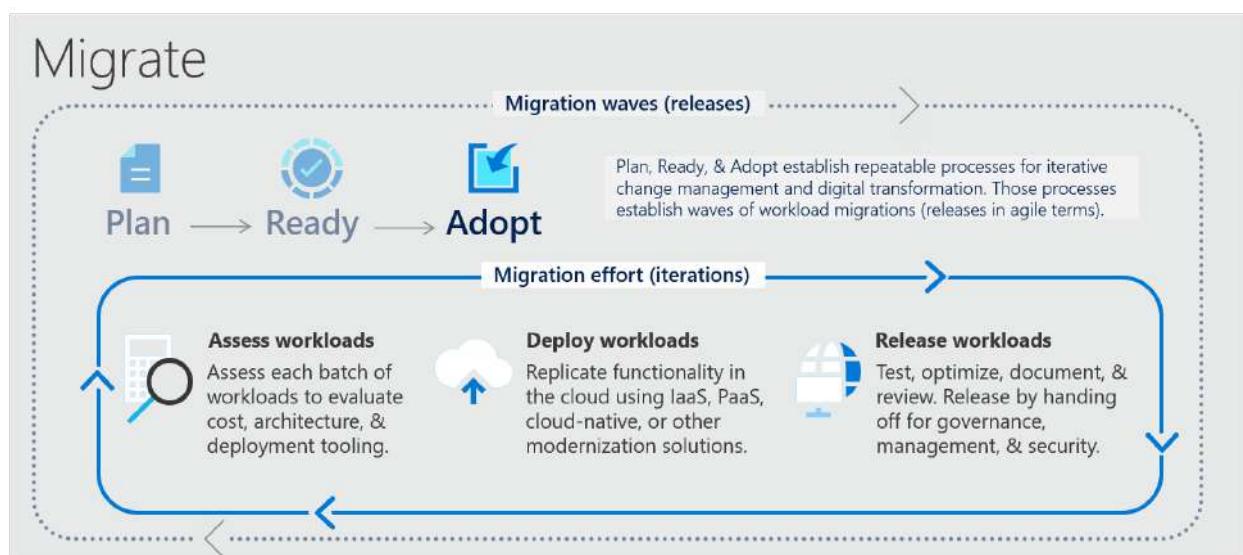


Figure 1

Planning and preparing your environment for Azure VMware Solution deployment is critical for a successful migration. Use your documented plan as a reference during deployment and migration, and make sure you've created a landing zone to host the workloads you plan to build in, or migrate to the cloud. A successful deployment results in a production-ready environment for creating or migrating Azure VMware Solution.

Azure VMware Solution process details

There are data points outside of a standard Azure Migrate assessment that you need to prepare for build out and migration.

- Identify the Azure subscription, resource group, region, and resource name
- Identify the number hosts required for deployment

- Request a host quota for a subscription with an eligible Azure plan
- Identify a minimum of a non-overlapping /22 CIDR IP segment for private cloud management
- Identify or deploy a single Azure Virtual Network
- Deploy the virtual network gateway in the virtual network to peer with the Azure VMware Solution ExpressRoute
- Define VMware NSX-T Data Center network segments for various administrative tasks like vMotioning servers from on-premises vSphere into Azure VMware Solution

From a technical point of view, it's important to get the core foundations right around networking and Migrate methodology.

- Use the right IP address space for networking will ensure that you can move workloads seamlessly between on-premises vSphere and Azure VMware Solution.
- Plan your Migrate methodology up front: live, cold, and bulk.
- Define the correct firewall rules: the right ports need to be open between on-premises vSphere and Azure VMware Solution before you deploy the service.

Azure VMware Solution process flow

Microsoft has a network of Azure VMware Solution certified partners. These partners can help you assess, deploy, and migrate your on-premises workloads into Azure VMware Solution.

You can deploy Azure VMware Solution by using one of the following options:

- Azure portal, where Azure VMware Solution can be deployed like any other service
- Azure command line interface
- Azure Resource Manager template, Bicep template, or Terraform template

When you deploy Azure VMware Solution, you get a software-defined data center that has vCenter Server, ESXi, vSAN, and NSX-T Data Center deployed. As a result, you can migrate workloads from your on-premises vSphere environments, deploy new virtual machines (VMs) within Azure VMware Solution, and consume Azure services from your private clouds. Everything you need to set up a successful migration, transformation, data center extension, is included when you deploy Azure VMware Solution.

Example migration process

Azure VMware Solution lets you seamlessly move VMware vSphere workloads from your data center to Azure and integrate more Azure services with ease. All while you continue

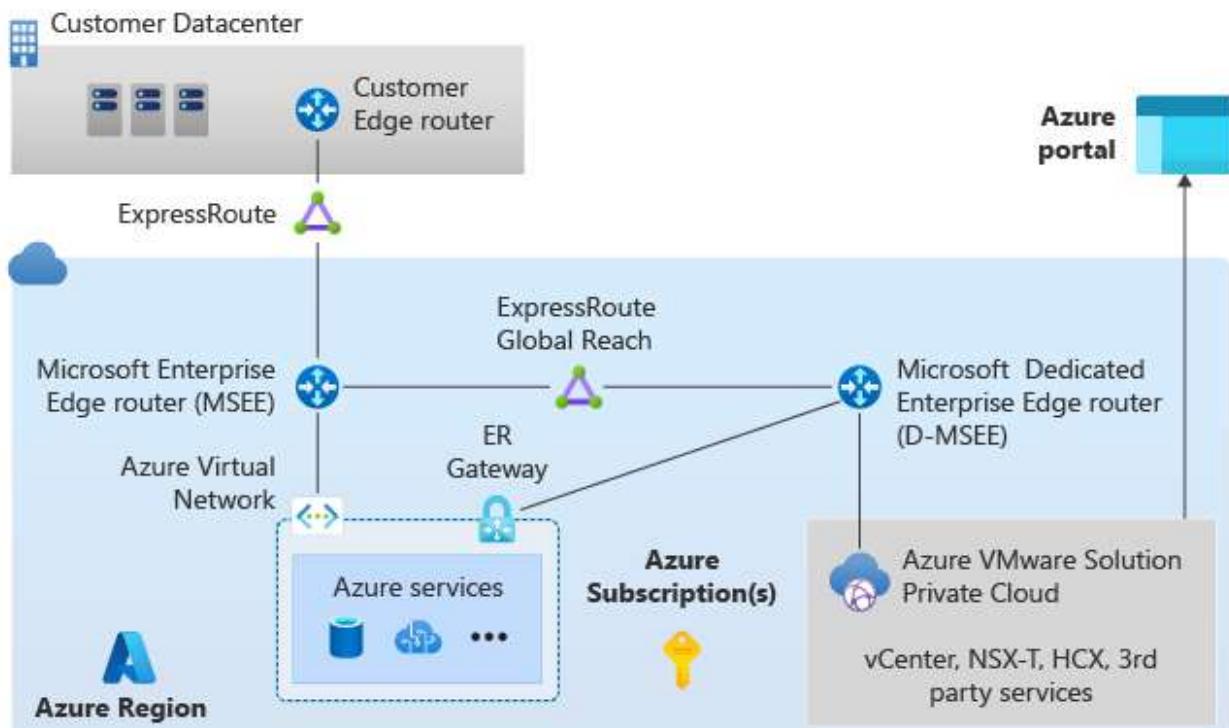
to manage your IT environments with the same VMware solution tools you already know. You have the choice and flexibility to determine what workloads to migrate, and you decide the right time to migrate them. With platform symmetry, you have complete control to transform based on how your organization defines its unique cloud journey.

You can take the tooling and operational best practices that you're already using and repurpose them in Azure with the Azure VMware Solution platform.

The hardware and software specifications should be familiar if you're a VMware vSphere administrator. If you're deploying Azure VMware Solution, it should match up to the VMware vSphere solution that you have on-premises, or maybe it's a version ahead.

What's important to note is that you'll need a minimum of three nodes per vSphere cluster. There's a maximum of 16 nodes per vSphere cluster, and then a maximum of 96 nodes in an Azure VMware Solution private cloud instance.

When you deploy Azure VMware Solution, at a minimum, you get three nodes and an ExpressRoute circuit. Because your Azure VMware Solution environment is deployed on bare metal servers, it needs to be peered into Azure for network connectivity. After you deploy Azure VMware Solution, you'll peer the ExpressRoute into an Azure virtual network. Then you can enable Global Reach between the Azure VMware Solution ExpressRoute and on-premises ExpressRoute circuits. Global Reach handles the east-west traffic routing between the two circuits using BGP. This is how you can think about migrating your VMs from on-premises vSphere all the way into the Azure VMware Solution private cloud.



After you deploy Azure VMware Solution, it will look like any other Azure service in the Azure portal. But when you're building the service, you'll need to provide a management IP address, which is different than the virtual network you're connecting the environment to. The management IP address requires a minimum of /22 CIDR block. You don't have to worry about subnetting your environment, Azure VMware Solution will do it for you. You can also enable the internet. This is where you might think about using Azure Virtual Network and Application Gateway.

Once you migrate your on-premises VMware vSphere environment into Azure, you have the opportunity to be closer to the Azure Resource Manager APIs, versus trying to think about it in a hybrid scenario. On the ExpressRoute circuit, you request your authorization keys, but some of the configuration is done for you. You'll also need to configure the HCX environment. That's what does the migrations from on-premises vSphere into Azure. Then configure your public IP and the ExpressRoute Global Reach.

The vCenter Server credentials and the NSX-T Manager credentials are set up for you during deployment. You can add segments to the NSX-T Data Center, and setup DHCP or DNS if you need it.

You can think about Azure VMware Solution as VMware Software-Defined Data Center as a service. The offering is between infrastructure as a service (IaaS) and platform as a service (PaaS). It's not one over the other. You can configure a jump host within the environment to access Azure VMware Solution. The jump host can be behind an Azure Bastion resource or configured with a public IP and just in time access. The Azure Bastion host is a way to provide secure access into that VM without having to expose the RDP port on a public IP address. A VM with just in time configured allows administrators to access an environment on a timed basis, so the RDP port isn't exposed and not a security vulnerability. Configuring everything in this manner allows access to your VM if there's ever an issue with ExpressRoute circuit coming from on-premises vSphere into Azure.

When you go into your Azure VMware Solution environment, it should look just like it did when it was running on-premises. You can also integrate with Azure. For example, you can build content libraries on Azure Blob Storage so when you template your VMs, you can spin up new VMs quickly.

You can use the vSphere Client interface to manage vCenter Server and VMware's PowerShell command line interface plugin (PowerCLI).

Azure VMware Solution allows you to work in a familiar environment, with familiar tools. It allows you to iterate, innovate, and modernize at your own pace as you work through your digital transformation.

To get more experience migrating VMware vSphere with Azure VMware Solution, try one of these [hands on labs](#).

- **Azure VMware Solution private cloud deployment and connectivity:** A guided lab for VMware vSphere administrators, showing how to set up the cloud deployment and connectivity.
- **Azure VMware Solution workload migration with VMware HCX:** This lab addresses the migration components. It walks through how to create the environment, deploy the environment, and migrate VMs.

More resources:

- [Azure VMware Solution documentation](#)
- [Learning path: Run VMware workloads on Azure VMware Solution](#)

Azure VMware Solution workload specific activities

An Azure Migrate assessment will provide a way for you to analyze all workloads running in an on-premises VMware vSphere environment. Running the assessment over a period of 30 days (or longer) will provide an opportunity to right size the Azure VMware Solution node size deployment. Additionally, it will help you prioritize the flow for your production migration.

Next steps

Review how you can extend your governance approach across the Azure VMware Solution. Evaluate and manage risk tolerance by identifying high-risk areas for business, convert risk vectors into governing corporate policies, and extend governance policies across Cost Management, Security Baseline, Identity Baseline, Resource Consistency, and Deployment Acceleration disciplines.

[Govern Azure VMware Solution](#)

Build migration plan with Azure Migrate

04/23/2025

Follow this article to build your migration plan to Azure with [Azure Migrate](#).

Define cloud migration goals

Before you start, understanding and evaluating your [motivation](#) for moving to the cloud can contribute to a successful business outcome. As explained in the [Cloud Adoption Framework](#), there are a number of triggers and outcomes.

 [Expand table](#)

Business event	Migration outcome
Datacenter exit	Cost
Merger, acquisition, or divestiture	Reduction in vendor/technical complexity
Reduction in capital expenses	Optimization of internal operations
End of support for mission-critical technologies	Increase in business agility
Response to regulatory compliance changes	Preparation for new technical capabilities
New data sovereignty requirements	Scaling to meet market demands
Reduction in disruptions, and IT stability improvements	Scaling to meet geographic demands

Identifying your motivation helps you to pin down your strategic migration goals. The next step is to identify and plan a migration path that's tailored for your workloads. The [Azure Migrate: Discovery and Assessment](#) tool helps you to assess on-premises workloads, and provides guidance and tools to help you migrate.

Understand your digital estate

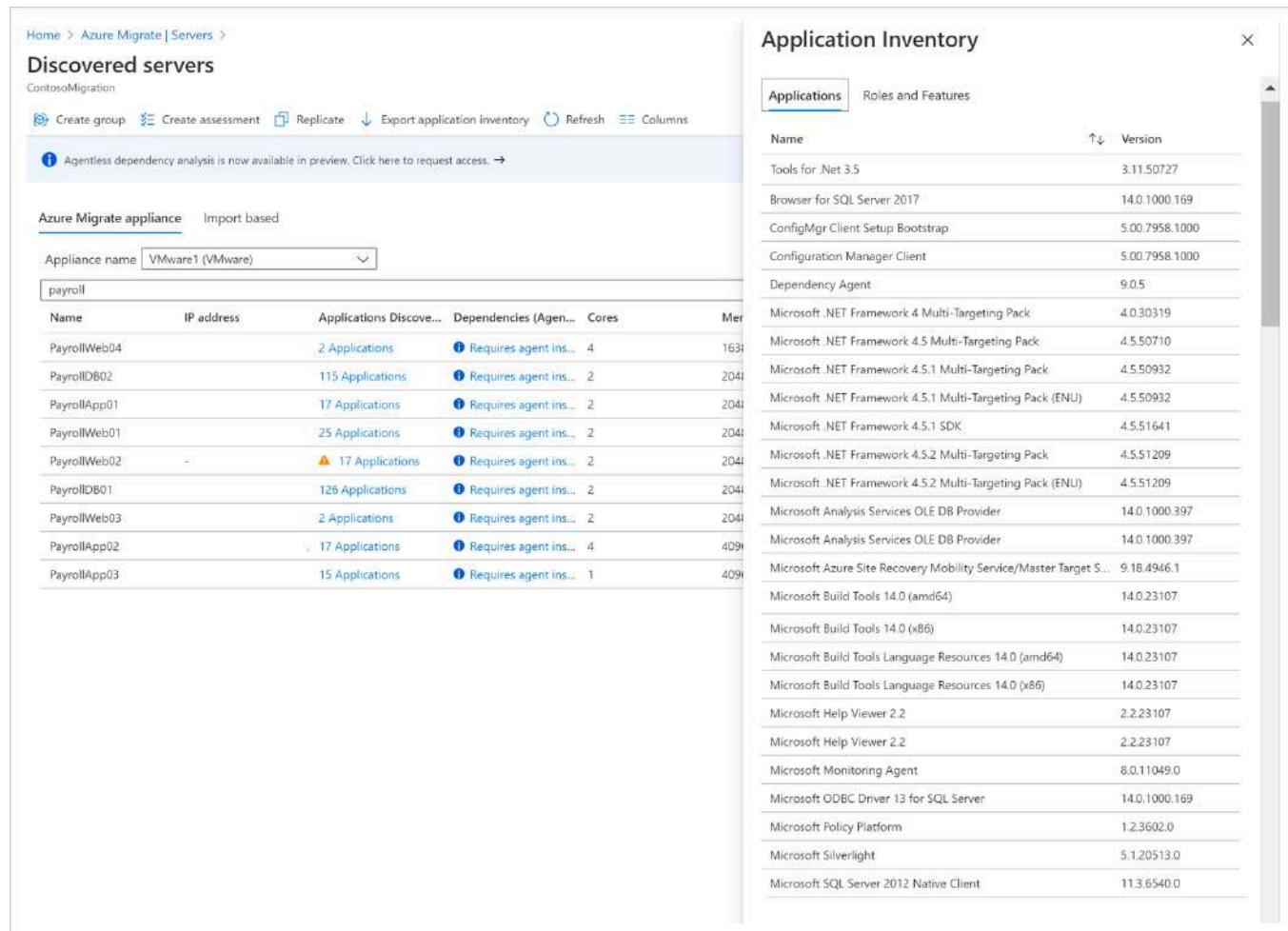
Start by identifying your on-premises infrastructure, applications, and dependencies. This helps you to identify workloads for migration to Azure, and to gather optimized cost projections. The Discovery and assessment tool helps you to identify the workloads you have in use, dependencies between workloads, and workload optimization.

Workloads in use

Azure Migrate uses a lightweight Azure Migrate appliance to perform agentless discovery of on-premises VMware VMs, Hyper-V VMs, other virtualized servers, and physical servers. Continuous discovery collects server configuration information, and performance metadata, and application data. Here's what the appliance collects from on-premises servers:

- Server, disk, and NIC metadata.
- Installed applications, roles, and features.
- Performance data, including CPU and memory utilization, disk IOPS, and throughput.

After collecting data, you can export the application inventory list to find apps, and SQL Server instances running on your servers. You can use the Azure Migrate: Database Assessment tool to understand SQL Server readiness.



The screenshot shows the Azure Migrate interface with two main sections: 'Discovered servers' and 'Application Inventory'.

Discovered servers: This section shows a list of servers discovered by the Azure Migrate appliance. The list includes:

Name	IP address	Applications Discove...	Dependencies (Agen...	Cores	Mem
PayrollWeb04		2 Applications	1 Requires agent ins...	4	1631
PayrollDB02		115 Applications	1 Requires agent ins...	2	2041
PayrollApp01		17 Applications	1 Requires agent ins...	2	2041
PayrollWeb01		25 Applications	1 Requires agent ins...	2	2041
PayrollWeb02	-	17 Applications	1 Requires agent ins...	2	2041
PayrollDB01		126 Applications	1 Requires agent ins...	2	2041
PayrollWeb03		2 Applications	1 Requires agent ins...	2	2041
PayrollApp02		17 Applications	1 Requires agent ins...	4	4091
PayrollApp03		15 Applications	1 Requires agent ins...	1	4091

Application Inventory: This section shows a list of applications and their details. The list includes:

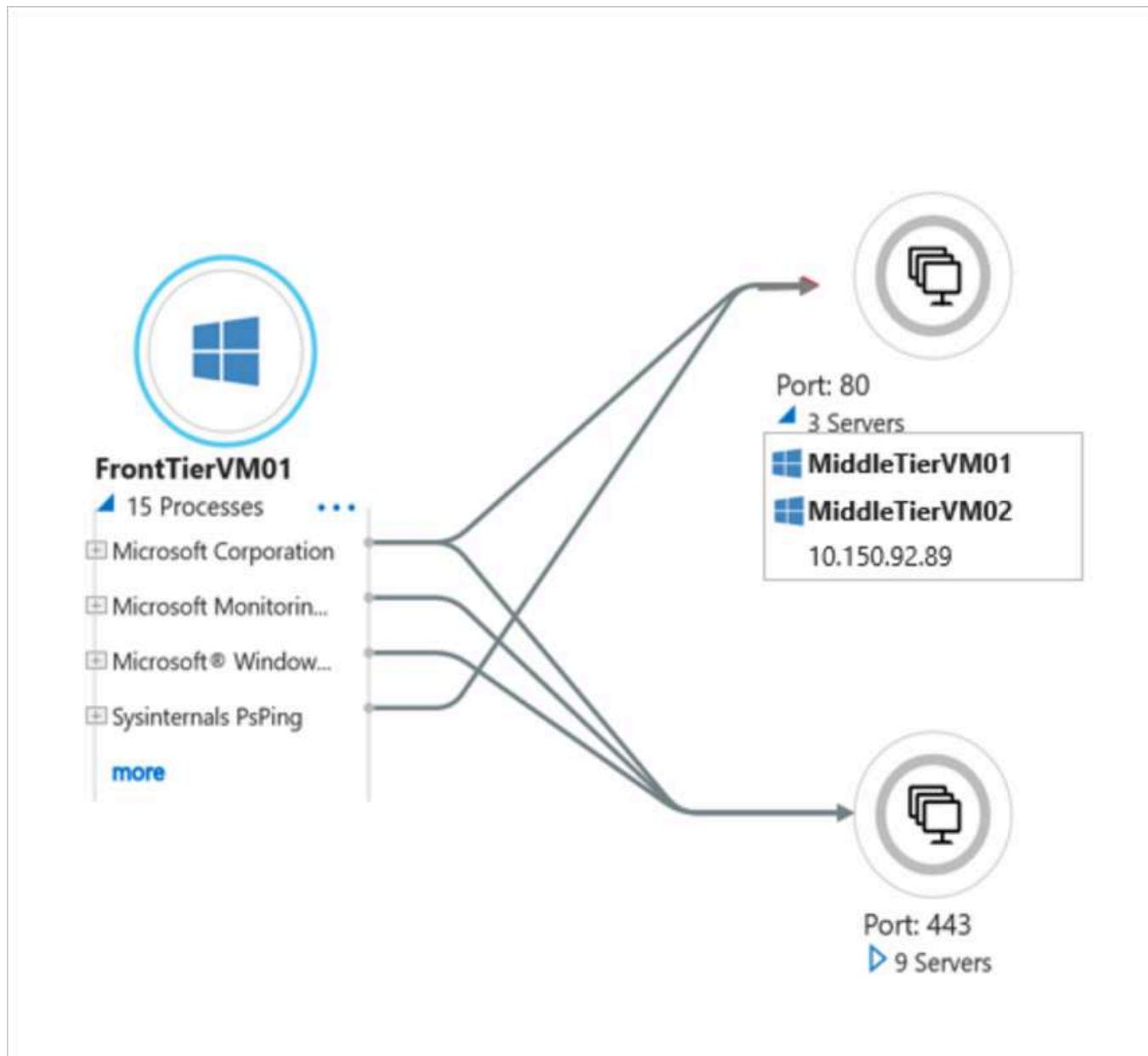
Name	Version
Tools for .Net 3.5	3.11.50727
Browser for SQL Server 2017	14.0.1000.169
ConfigMgr Client Setup Bootstrap	5.00.7958.1000
Configuration Manager Client	5.00.7958.1000
Dependency Agent	9.0.5
Microsoft .NET Framework 4 Multi-Targeting Pack	4.0.30319
Microsoft .NET Framework 4.5 Multi-Targeting Pack	4.5.50710
Microsoft .NET Framework 4.5.1 Multi-Targeting Pack	4.5.50932
Microsoft .NET Framework 4.5.1 Multi-Targeting Pack (ENU)	4.5.50932
Microsoft .NET Framework 4.5.1 SDK	4.5.51641
Microsoft .NET Framework 4.5.2 Multi-Targeting Pack	4.5.51209
Microsoft .NET Framework 4.5.2 Multi-Targeting Pack (ENU)	4.5.51209
Microsoft Analysis Services OLE DB Provider	14.0.1000.397
Microsoft Analysis Services OLE DB Provider	14.0.1000.397
Microsoft Azure Site Recovery Mobility Service/Master Target S...	9.18.4946.1
Microsoft Build Tools 14.0 (amd64)	14.0.23107
Microsoft Build Tools 14.0 (x86)	14.0.23107
Microsoft Build Tools Language Resources 14.0 (amd64)	14.0.23107
Microsoft Build Tools Language Resources 14.0 (x86)	14.0.23107
Microsoft Help Viewer 2.2	2.2.23107
Microsoft Help Viewer 2.2	2.2.23107
Microsoft Monitoring Agent	8.0.11049.0
Microsoft ODBC Driver 13 for SQL Server	14.0.1000.169
Microsoft Policy Platform	1.2.3602.0
Microsoft Silverlight	5.1.20513.0
Microsoft SQL Server 2012 Native Client	11.3.6540.0

	A	B	C	D	E	F	G
1	MachineName	Instance Name	Edition	Service Pack	Version		
2	FW12R2DSSP16-03	MSSQL12.MSSQLSERVER	Enterprise Edition: Core-based Licensing	1	12.1.4100.1		
3	FW12R2-SSP1602	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
4	FW12R2DSSP16-02	MSSQL12.MSSQLSERVER	Enterprise Edition: Core-based Licensing	1	12.1.4100.1		
5	FW12STR2S012-05	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
6	FW12STR2S012-06	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
7	PayrollDB02	MSSQL14.MSSQLSERVER	Enterprise Evaluation Edition	0	14.0.1000.169		
8	FW12STR2S012-01	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
9	FW12R2-SSP1302	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
10	FW12STR2S012-04	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
11	FW12R2DSSP13-05	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
12	FW12R2DSSP16-05	MSSQL12.MSSQLSERVER	Enterprise Edition: Core-based Licensing	1	12.1.4100.1		
13	FPL-VCEN55-01	MSSQL10.0.VIM_SQLEXP	Express Edition	1	10.51.2500.0		
14	FW12R2DSSP13-02	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
15	DMA VM	MSSQL12.MSSQLSERVER	Express Edition	0	11.0.2100.60		
16	FW12R2DSSP16-01	MSSQL12.MSSQLSERVER	Enterprise Edition: Core-based Licensing	1	12.1.4100.1		
17	FW12R2DSSP13-01	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
18	FPLV2A-SCOM16DB	MSSQL12.MSSQLSERVER	Enterprise Edition: Core-based Licensing	9	12.3.6024.0		
19	FW12R2-SSP1601	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
20	FWSTD2016-001	MSSQL13.MSSQLSERVER	Enterprise Edition: Core-based Licensing	0	13.0.1601.5		
21	PayrollDB01	MSSQL12.DATATIVERVM01	Express Edition	0	12.0.2000.8		
22	FW12R2DSSP13-04	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
23	FW12R2-SSP1603	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
24	FPLWIN2016-01	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	0	13.0.1601.5		
25	FW12R2DSSP13-03	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
26	FW12STR2S012-03	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
27	FW12R2-SSP1301	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	5	11.2.5058.0		
28	FW12R2DSSP16-04	MSSQL12.MSSQLSERVER	Enterprise Edition: Core-based Licensing	1	12.1.4100.1		
29	FW12R2LSP16D2-2	MSSQL11.MSSQLSERVER	Enterprise Edition	0	11.0.2100.60		
30	FW12STR2S012-02	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
31	FW12DCR2S012-10	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
32	FW12DCR2S012-03	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
33	FW12DCR2S012-08	MSSQL11.MSSQLSERVER	Enterprise Evaluation Edition	2	11.2.5058.0		
34	FW12DCR2S012-04	MSSQL11.MSSQLSERVER	Enterprise Evaluation Edition	2	11.2.5058.0		
35	FW12DCR2S012-13	MSSQL11.MSSQLSERVER	Enterprise Evaluation Edition	2	11.2.5058.0		
36	FW12DCS012-08	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
37	FW12DCR2S012-09	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
38	FW12DCR2S012-06	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
39	FW12DCR2S012-02	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
40	FW12DCS012-13	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		
41	FW12DCS012-11	MSSQL11.MSSQLSERVER	Enterprise Edition: Core-based Licensing	2	11.2.5058.0		

Along with data discovered with the Discovery and assessment tool, you can use your Configuration Management Database (CMDB) data to build a view of your server and database estate, and to understand how your servers are distributed across business units, application owners, geographies, etc. This helps decide which workloads to prioritize for migration.

Dependencies between workloads

After server discovery, you can [analyze dependencies](#), to visualize and identify cross-server dependencies, and optimization strategies for moving interdependent servers to Azure. The visualization helps to understand whether certain servers are in use, or if they can be decommissioned, instead of being migrated. Analyzing dependencies helps ensure that nothing is left behind, and to surprise outages during migration. With your application inventory and dependency analysis done, you can create high-confidence groups of servers, and start assessing them.



Optimization and sizing

Azure provides flexibility to resize your cloud capacity over time, and migration provides an opportunity for you to optimize the CPU and memory resources allocated to your servers. Creating an assessment on servers you've identity helps you to understand your workload performance history. This is crucial for right-sizing Azure VM SKUs, and disk recommendations in Azure.

Assess migration readiness

Readiness/suitability analysis

You can export the assessment report, and filter on these categories to understand Azure readiness:

- **Ready for Azure:** Servers can be migrated as-is to Azure, without any changes.
- **Conditionally ready for Azure:** Servers can be migrated to Azure, but need minor changes, in accordance with the remediation guidance provided in the assessment.
- **Not ready for Azure:** Servers can't be migrated to Azure as-is. Issues must be fixed in accordance with remediation guidance, before migration.
- **Readiness unknown:** Azure Migrate can't determine server readiness, because of insufficient metadata.

Using database assessments, you can assess the readiness of your SQL Server data estate for migration to Azure SQL Database, or Azure SQL Managed Instances. The assessment shows migration readiness status percentage for each of your SQL server instances. In addition, for each instance you can see the recommended target in Azure, potential migration blockers, a count of breaking changes, readiness for Azure SQL DB or Azure SQL VM, and a compatibility level. You can dig deeper to understand the impact of migration blockers, and recommendations for fixing them.

Assessed databases											
Columns		Subscription : DMSMigrationDemo		Migrate project : DBReadiness (DBReadiness)							
Microsoft tools											
Search to filter rows											
NAME	DATABASE SIZE (GB)	TARGET IN AZURE	MIGRATION BLOCKERS	BREAKING CHANGES COUNT	READY FOR AZURE SQL DB	READY FOR AZURE SQL VM					
Archive	0.003	Azure SQL Database Managed I...	1	0	No	Yes					
ADworks	0.02	Azure SQL Database Managed I...	0	0	Yes	Yes					
DBforTDE	0.003	Azure SQL Database Managed I...	0	0	Yes	Yes					
SitesEE	0.4	Azure SQL Database Managed I...	0	0	Yes	Yes					
HR	0.02	Azure SQL Database Managed I...	0	0	Yes	Yes					
Inventory	0.1	Azure SQL Database Managed I...	0	0	Yes	Yes					
StackOverflowEE	0.5	Azure SQL Database Managed I...	0	0	Yes	Yes					

Sizing Recommendations

After a server is marked as ready for Azure, Discovery and assessment makes sizing recommendations that identify the Azure VM SKU and disk type for your servers. You can get sizing recommendations based on performance history (to optimize resources as you migrate), or based on on-premises server settings, without performance history. In a database assessment, you can see recommendations for the database SKU, pricing tier, and compute level.

Get compute costs

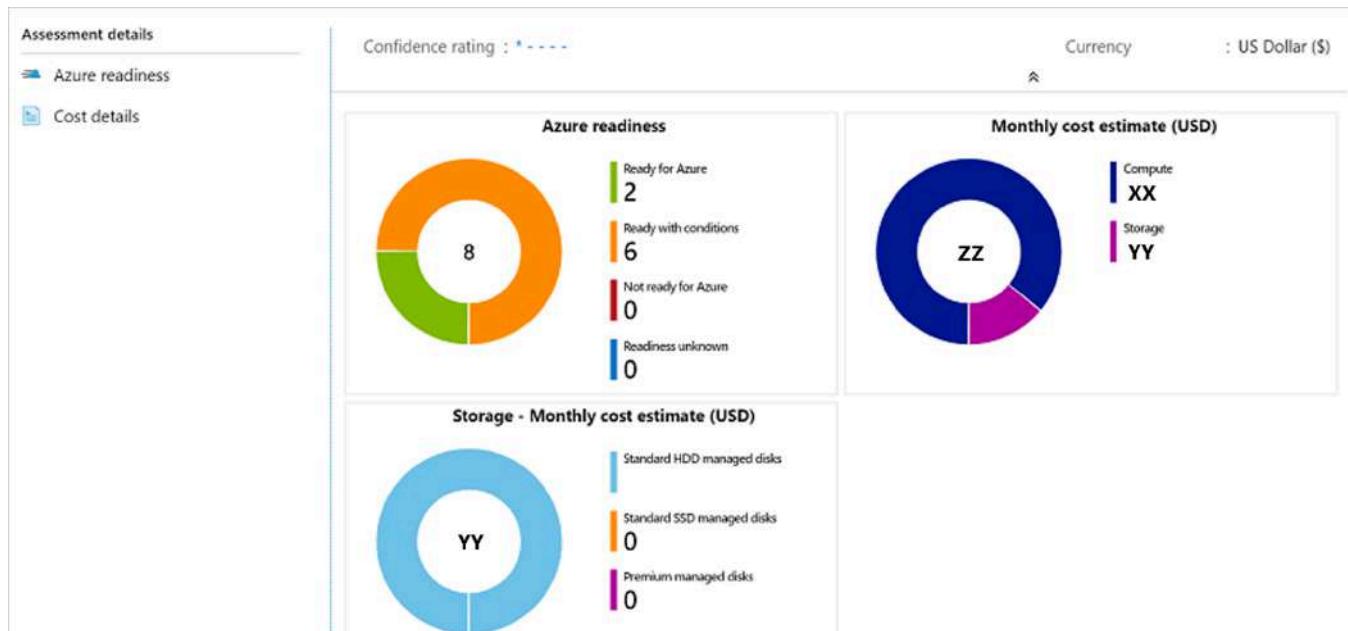
Performance-based sizing option in Azure Migrate assessments helps you to right-size VMs, and should be used as a best practice for optimizing workloads in Azure. In addition to right-

sizing, there are a few other options to help save Azure costs:

- **Reserved Instances:** With [reserved instances\(RI\)](#), you can significantly reduce costs compared to [pay-as-you-go pricing](#).
- **Azure Hybrid Benefit:** With [Azure Hybrid Benefit](#), you can bring on-premises Windows Server licenses with active Software Assurance, or Linux subscriptions, to Azure, and combine with reserved instances options.
- **Enterprise Agreement:** Azure [Enterprise Agreements \(EA\)](#) can offer savings for Azure subscriptions and services.
- **Offers:** There are multiple [Azure Offers](#). For example, [Pay-As-You-Go Dev/Test](#), or [Enterprise Dev/Test offer](#), to provide lower rates for dev/test VMs
- **VM uptime:** You can review days per month and hours per day in which Azure VMs run. Shutting off servers when they're not in use can reduce your costs (not applicable for RIs).
- **Target region:** You can create assessments in different regions, to figure out whether migrating to a specific region might be more cost effective.

Visualize data

You can view Discovery and assessment reports (with Azure readiness information, and monthly cost distribution) in the portal. You can also export assessment, and enrich your migration plan with additional visualizations. You can create multiple assessments, with different combinations of properties, and choose the set of properties that work best for your business.



Evaluate gaps/blockers

As you figure out the apps and workloads you want to migrate, identify downtime constraints for them, and look for any operational dependencies between your apps and the underlying

infrastructure. This analysis helps you to plan migrations that meet your recovery time objective (RTO), and ensure minimal to zero data loss. Before you migrate, we recommend that you review and mitigate any compatibility issues, or unsupported features that may block server/SQL database migration. The Azure Migrate Discovery and assessment report, and Azure Migrate Database Assessment, can help with this.

Prioritize workloads

After you've collected information about your inventory, you can identify which apps and workloads to migrate first. Develop an "apply and learn" approach to migrate apps in a systematic and controllable way, so that you can iron out any flaws before starting a full-scale migration.

To prioritize migration order, you can use strategic factors such as complexity, time-to-migrate, business urgency, production/non-production considerations, compliance, security requirements, application knowledge, etc.

A few recommendations:

- **Prioritize quick wins:** Use the assessment reports to identify low-hanging fruit, including servers and databases that are fully ready, and require minimal effort to migrate to Azure. The table summarizes a few ways to do this.

[] [Expand table](#)

State	Action
Azure ready VMs	Export the assessment report, and filter all servers with state <i>Ready for Azure</i> . This might be the first group of servers that you lift and shift to Azure, using the Migration and modernization tool.
End-of-support operating systems	Export the assessment report, and filter all servers running Windows Server 2008 R2/Windows Server 2008. These operating systems are at the end of support, and only Azure provides a free three years of security updates when you migrate them to Azure. If you combine Azure Hybrid Benefit, and use RIs, the savings could be higher.
SQL Server migration	Use the database assessment recommendations to migrate databases that are ready for Azure SQL Database, using the Azure Migrate: Database Migration tool. Migrate the databases ready for Azure SQL VM using the Migration and modernization tool.
End-of-support software	Export your application inventory, and filter for any software/extensions that might be reaching end-of-support. Prioritize these applications for migration.

State	Action
Under-provisioned servers	Export the assessment report, and filter for servers with low CPU utilization (%) and memory utilization (%). Migrate to a right-sized Azure VM, and save on costs for underutilized resources.
Over-provisioned servers	Export the assessment report and filter for servers with high CPU utilization (%) and memory utilization (%). Solve capacity constraints, prevent overstrained servers from breaking, and increase performance by migrating these servers to Azure. In Azure, use autoscaling capabilities to meet demand.
	Analyze assessment reports to investigate storage constraints. Analyze disk IOPS and throughput, and the recommended disk type.

- **Start small, then go big:** Start by moving apps and workloads that present minimal risk and complexity, to build confidence in your migration strategy. Analyze Azure Migrate assessment recommendations together with your CMDB repository, to find and migrate dev/test workloads that might be candidates for pilot migrations. Feedback and learnings from pilot migrations can be helpful as you begin migrating production workloads.
- **Comply:** Azure maintains the largest compliance portfolio in the industry, in terms of breadth and depth of offerings. Use compliance requirements to prioritize migrations, so that apps and workloads comply with your national/regional and industry-specific standards and laws. This is especially true for organizations that deal with business-critical process, hold sensitive information, or are in heavily regulated industries. In these types of organizations, standards and regulations abound, and might change often, being difficult to keep up with.

Finalize the migration plan

Before finalizing your migration plan, make sure you consider and mitigate other potential blockers, as follows:

- **Network requirements:** Evaluate network bandwidth and latency constraints, which might cause unforeseen delays and disruptions to migration replication speed.
- **Testing/post-migration tweaks:** Allow a time buffer to conduct performance and user acceptance testing for migrated apps, or to configure/tweak apps post-migration, such as updating database connection strings, configuring web servers, performing cut-overs/cleanup etc.
- **Permissions:** Review recommended Azure permissions, and server/database access roles and permissions needed for migration.
- **Training:** Prepare your organization for the digital transformation. A solid training foundation is important for successful organizational change. Check out free [Microsoft](#)

[Learn training](#), including courses on Azure fundamentals, solution architectures, and security. Encourage your team to explore [Azure certifications](#).

- **Implementation support:** Get support for your implementation if you need it. Many organizations opt for outside help to support their cloud migration. To move to Azure quickly and confidently with personalized assistance, consider an [Azure Expert Managed Service Provider](#), or [FastTrack for Azure](#).

Create an effective cloud migration plan that includes detailed information about the apps you want to migrate, app/database availability, downtime constraints, and migration milestones.

The plan considers how long the data copy takes, and include a realistic buffer for post-migration testing, and cut-over activities.

A post-migration testing plan should include functional, integration, security, and performance testing and use cases, to ensure that migrated apps work as expected, and that all database objects, and data relationships, are transferred successfully to the cloud.

Build a migration roadmap, and declare a maintenance window to migrate your apps and databases with minimal to zero downtime, and limit the potential operational and business impact during migration.

Migrate

We recommend that you run a test migration in Azure Migrate, before starting a full-scale migration. A test migration helps you to estimate the time involved, and tweak your migration plan. It provides an opportunity to discover any potential issues, and fix them before the full migration.

When you're ready for migration, use the Migration and modernization tool, and the Azure Data Migration Service (DMS), for a seamless and integrated migration experience, with end-to-end tracking.

- With the Migration and modernization tool, you can migrate on-premises VMs and servers, or VMs located in other private or public cloud (including AWS, GCP) with around zero downtime.
- Azure DMS provides a fully managed service that's designed to enable seamless migrations from multiple database sources to Azure Data platforms, with minimal downtime.

Upgrade Windows OS

Azure Migrate provides an option to customers to upgrade their Windows Server OS seamlessly during the migration. Azure Migrate OS upgrade allows you to move from an older

operating system to a newer one while keeping your settings, server roles, and data intact.

[Learn more](#).

Azure Migrate OS upgrade uses an Azure VM [Custom script extension](#) to perform the following activities for an in-place upgrade experience:

- A data disk containing Windows Server setup files is created and attached to the VM.
- A Custom Script Extension called `InPlaceOsUpgrade` is enabled on the VM, which downloads a script from the storage account and initiates the upgrade in a quiet mode.

Next steps

- Investigate the [cloud migration journey](#) in the Azure Cloud Adoption Framework.
- Get a [quick overview](#) of Azure Migrate, and watch a [getting started video](#).
- Learn more about assessing VMs for migration to [Azure VMs](#).

Support matrix for VMware vSphere migration

05/13/2025

This article summarizes support settings and limitations for migrating VMware vSphere VMs with [Migration and modernization](#). If you're looking for information about assessing VMware vSphere VMs for migration to Azure, review the [assessment support matrix](#).

Caution

This article references CentOS, a Linux distribution that is End Of Life (EOL) status. Please consider your use and planning accordingly. For more information, see the [CentOS End Of Life guidance](#).

Caution

This article references Windows Server versions that have reached End of Support (EOS). Microsoft has officially ended support for the following operating systems:

- Windows Server 2003
- Windows Server 2008 (including SP2 and R2 SP1)
- Windows Server 2012
- Windows Server 2012 R2 As a result, Azure Migrate doesn't guarantee consistent or reliable outcomes for these OS versions. Customers may face problems and are strongly advised to upgrade to a supported Windows Server version before starting migration.

Migration options

You can migrate VMware vSphere VMs in a couple of ways:

- **Using agentless migration:** Migrate VMs without needing to install anything on them. You deploy the [Azure Migrate appliance](#) for agentless migration.
- **Using agent-based migration:** Install an agent on the VM for replication. For agent-based migration, you deploy a [replication appliance](#).

Note

This also supports migrating VMs from AVS.

Review [this article](#) to figure out which method you want to use.

Agentless migration

This section summarizes requirements for agentless VMware vSphere VM migration to Azure.

VMware vSphere requirements (agentless)

The VMware vSphere hypervisor requirements are:

- **VMware vCenter Server** - Version 8.0 & subsequent updates in this version, Version 7.0, 6.7 or 6.5.
- **VMware vSphere ESXi host** - Version 8.0 & subsequent updates in this version, Version 7.0, 6.7 or 6.5.
- **Multiple vCenter Servers** - A single appliance can connect to up to 10 vCenter Servers.
- **vCenter Server permissions** - The VMware account used to access the vCenter server from the Azure Migrate appliance must have the following permissions assigned at all required levels - datacenter, cluster, host, VM, and datastore. Ensure permissions are applied at each level to avoid replication errors.

 [Expand table](#)

Privilege Name in the vSphere Client	The purpose for the privilege	Required On	Privilege Name in the API
Browse datastore	Allow browsing of VM log files to troubleshoot snapshot creation and deletion.	Data stores	Datastore.Browse
Low level file operations	Allow read/write/delete/rename operations in the datastore browser to troubleshoot snapshot creation and deletion.	Data stores	Datastore.FileManagement

Privilege Name in the vSphere Client	The purpose for the privilege	Required On	Privilege Name in the API
Change Configuration - Toggle disk change tracking	Allow enable or disable change tracking of VM disks to pull changed blocks of data between snapshots.	Virtual machines	VirtualMachine.Config.ChangeTracking
Change Configuration - Acquire disk lease	Allow disk lease operations for a VM to read the disk using the VMware vSphere Virtual Disk Development Kit (VDDK).	Virtual machines	VirtualMachine.Config.DiskLease
Provisioning - Allow read-only disk access	Allow read-only disk access: Allow opening a disk on a VM to read the disk using the VDDK.	Virtual machines	VirtualMachine.Provisioning.DiskRandomRead
Provisioning - Allow disk access	Allow opening a disk on a VM to read the disk using the VDDK.	Virtual machines	VirtualMachine.Provisioning.DiskRandomAccess
Provisioning - Allow virtual machine download	Allow virtual machine download: Allows read operations on files associated with a VM to download the logs and troubleshoot if failure occurs.	Root host or vCenter Server	VirtualMachine.Provisioning.GetVmFiles
Snapshot management	Allow creation and management of VM snapshots for replication.	Virtual machines	VirtualMachine.GuestOperations.*
Guest operations	Allow Discovery, Software Inventory, and Dependency Mapping on VMs.	Virtual machines	VirtualMachine.State.*
Interaction Power Off	Allow the VM to be powered off during migration to Azure.	Virtual machines	VirtualMachine.Interact.PowerOff

VM requirements (agentless)

The table summarizes agentless migration requirements for VMware vSphere VMs.

! Note

If a major version of an operating system is supported in agentless migration, all minor versions and kernels are automatically supported.

[Expand table](#)

Support	Details
Supported operating systems	<p>Windows Server 2003 and later versions. Learn more.</p> <p>You can migrate all the Linux operating systems supported by Azure listed here.</p>
Windows VMs in Azure	<p>You might need to make some changes on VMs before migration.</p>
Linux VMs in Azure	<p>Some VMs might require changes so that they can run in Azure.</p> <p>For Linux, Azure Migrate makes the changes automatically for these operating systems:</p> <ul style="list-style-type: none">- Red Hat Enterprise Linux 9.5, 9.x, 8.x, 7.9, 7.8, 7.7, 7.6, 7.5, 7.4, 7.3, 7.2, 7.1, 7.0, 6.x- CentOS Stream- SUSE Linux Enterprise Server 15 SP6, 15 SP5, 15 SP4, 15 SP3, 15 SP2, 15 SP1, 15 SP0, 12, 11 SP4, 11 SP3- Ubuntu 24.04, 22.04, 21.04, 20.04, 19.04, 19.10, 18.04LTS, 16.04LTS, 14.04LTS- Debian 11, 10, 9, 8, 7- Oracle Linux 9, 8, 7.7-Cl, 7.7, 6- Kali Linux (2016, 2017, 2018, 2019, 2020, 2021, 2022)- Alma Linux 8.x, 9.x- Rocky Linux 8.x, 9.x <p>For other operating systems, you make the required changes manually. The <code>SELinux Enforced</code> setting is currently not fully supported. It causes Dynamic IP setup and Microsoft Azure Linux Guest agent (waagent/WALinuxAgent) installation to fail. You can still migrate and use the VM. The <code>SELinux Permissive</code> setting is supported.</p>
Boot requirements	<p>Windows VMs:</p> <p>OS Drive (C:\) and System Reserved Partition (EFI System Partition for UEFI VMs) should reside on the same disk.</p> <p>If <code>/boot</code> is on a dedicated partition, it should reside on the OS disk and not be spread across multiple disks.</p> <p>If <code>/boot</code> is part of the root (/) partition, then the '/' partition should be on the OS disk and not span other disks.</p> <p>Linux VMs:</p>

Support	Details
	<p>If <code>/boot</code> is on a dedicated partition, it should reside on the OS disk and not be spread across multiple disks.</p> <p>If <code>/boot</code> is part of the root (/) partition, then the '/' partition should be on the OS disk and not span other disks.</p>
UEFI boot	UEFI-based virtual machines are migrated to Azure's Generation 2 VMs. However, it's important to note that Azure Generation 2 VMs lack the Secure Boot feature. For VMs that utilized Secure Boot in their original configuration, a conversion to Trusted Launch VMs is recommended after migration. This step ensures that Secure Boot, along with other enhanced security functionalities, is re-enabled.
Disk size	Up to 2TB OS disk for Gen1 VM and up to 4TB OS disk for Gen2 VM; 32 TB for data disks. Changing the size of the source disk after initiating replication is supported and won't impact ongoing replication cycle.
Dynamic disk	<ul style="list-style-type: none"> - An OS disk as a dynamic disk isn't supported. - If a VM with OS disk as dynamic disk is replicating, convert the disk type from dynamic to basic and allow the new cycle to complete, before triggering test migration or migration. You'll need help from OS support for conversion of dynamic to basic disk type.
Ultra disk	Ultra disk migration isn't supported from the Azure Migrate portal. You have to do an out-of-band migration for the disks that are recommended as Ultra disks. That is, you can migrate selecting it as premium disk type and change it to Ultra disk after migration.
Encrypted disks/volumes	VMs with encrypted disks/volumes aren't supported for migration.
Shared disk cluster	Not supported.
Independent disks	Not supported.
RDM/passthrough disks	If VMs have RDM or passthrough disks, these disks won't be replicated to Azure.
NFS	NFS volumes mounted as volumes on the VMs won't be replicated.
ReiserFS	Not supported.
iSCSI targets	VMs with iSCSI targets aren't supported for agentless migration.
Multipath IO	Not supported.
Storage vMotion	Supported.
Teamed NICs	Not supported.
IPv6	Not supported.
NVMe disks	Not supported.

Support	Details
Target disk	VMs can be migrated only to managed disks (standard HDD, standard SSD, Premium V2 SSD (preview) for data disks) in Azure.
Simultaneous replication	Up to 300 simultaneously replicating VMs per vCenter Server with one appliance. Up to 500 simultaneously replicating VMs per vCenter Server when an additional scale-out appliance is deployed.
Automatic installation of Azure VM agent (Windows and Linux Agent)	<p>Windows:</p> <p>Supported for Windows Server 2008 R2 onwards.</p> <p>Linux:</p> <ul style="list-style-type: none"> - Red Hat Enterprise Linux 9.x, 8.x, 7.9, 7.8, 7.7, 7.6, 7.5, 7.4, 7.0, 6.x - CentOS Stream - SUSE Linux Enterprise Server 15 SP6, 15 SP5, 15 SP4, 15 SP3, 15 SP2, 15 SP1, 15 SP0, 12, 11 SP4, 11 SP3 - Ubuntu 22.04, 21.04, 20.04, 19.04, 19.10, 18.04LTS, 16.04LTS, 14.04LTS - Debian 11, 10, 9, 8, 7 - Oracle Linux 9, 8, 7.7-Cl, 7.7, 6 - Kali Linux (2016, 2017, 2018, 2019, 2020, 2021, 2022)

(!) Note

Ensure that the following special characters are not passed in any credentials as they are not supported for SSO passwords:

- Non-ASCII characters. [Learn more](#) ↗.
- Ampersand (&)
- Semicolon (;)
- Double quotation mark (")
- Single quotation mark (')
- Circumflex (^)
- Backslash (\)
- Percentage (%)
- Angle brackets (<,>)
- Pound (£)

(!) Note

In addition to the Internet connectivity, for Linux VMs, ensure that the following packages are installed for successful installation of Microsoft Azure Linux agent (waagent):

- Python 2.6+
- OpenSSL 1.0+
- OpenSSH 5.3+
- Filesystem utilities: sfdisk, fdisk, mkfs, parted
- Password tools: chpasswd, sudo
- Text processing tools: sed, grep
- Network tools: ip-route

💡 Tip

Using the Azure portal you'll be able to select up to 10 VMs at a time to configure replication. To replicate more VMs you can use the portal and add the VMs to be replicated in multiple batches of 10 VMs, or use the Azure Migrate PowerShell interface to configure replication. Ensure that you don't configure simultaneous replication on more than the maximum supported number of VMs for simultaneous replications.

Appliance requirements (agentless)

Agentless migration uses the [Azure Migrate appliance](#). You can deploy the appliance as a VMware vSphere VM using an OVA template, imported into vCenter Server, or using a [PowerShell script](#).

- Learn about [appliance requirements](#) for VMware vSphere.
- Learn about URLs that the appliance needs to access in [public](#) and [government](#) clouds.
- In Azure Government, you must deploy the appliance [using the script](#)s

Port requirements (agentless)

expand Expand table

Device	Connection
Appliance	Outbound connections on port 443 to upload replicated data to Azure, and to communicate with Azure Migrate services orchestrating replication and migration.
vCenter Server	Inbound connections on port 443 to allow the appliance to orchestrate replication - create snapshots, copy data, release snapshots.
vSphere ESXi host	Inbound on TCP port 902 for the appliance to replicate data from snapshots. Outbound on port 902 from ESXi host is required for sending heartbeat traffic to vCenter

Agent-based migration

This section summarizes requirements for agent-based migration.

VMware vSphere requirements (agent-based)

This table summarizes assessment support and limitations for VMware vSphere virtualization servers.

[Expand table](#)

VMware vSphere requirements	Details
VMware vCenter Server	Version 8.0 & subsequent updates in this version, Version 7.0, 6.7 or 6.5..
VMware vSphere ESXi host	Version 8.0 & subsequent updates in this version, Version 7.0, 6.7 or 6.5..
vCenter Server permissions	<p>VM discovery: At least a read-only user Data Center object -> Propagate to Child Object, role=Read-only.</p> <p>Replication: Create a role (Azure Site Recovery) with the required permissions, and then assign the role to a VMware vSphere user or group Data Center object -> Propagate to Child Object, role=Azure Site Recovery</p> <p>Datastore -> Allocate space, browse datastore, low-level file operations, remove file, update virtual machine files</p> <p>Network -> Network assign</p> <p>Resource -> Assign VM to resource pool, migrate powered off VM, migrate powered on VM</p> <p>Tasks -> Create task, update task</p> <p>Virtual machine -> Configuration</p> <p>Virtual machine -> Interact -> answer question, device connection, configure CD media, configure floppy media, power off, power on, VMware tools install</p> <p>Virtual machine -> Inventory -> Create, register, unregister</p> <p>Virtual machine -> Provisioning -> Allow virtual machine download, allow virtual machine files upload</p>

VMware vSphere requirements	Details
	<p>Virtual machine -> Snapshots -> Remove snapshots.</p> <p>Note: User assigned at datacenter level, and has access to all the objects in the datacenter.</p> <p>To restrict access, assign the No access role with the Propagate to child object, to the child objects (vSphere hosts, datastores, VMs, and networks).</p>

VM requirements (agent-based)

The table summarizes VMware vSphere VM support for VMware vSphere VMs you want to migrate using agent-based migration.

[Expand table](#)

Support	Details
Machine workload	Azure Migrate supports migration of any workload (say Active Directory, SQL server, etc.) running on a supported machine.
Operating systems	For the latest information, review the operating system support for Site Recovery. Azure Migrate provides identical VM operating system support.
Linux file system/guest storage	For the latest information, review the Linux file system support for Site Recovery. Azure Migrate has identical Linux file system support.
Network/Storage	For the latest information, review the network and storage prerequisites for Site Recovery. Azure Migrate provides identical network/storage requirements.
Azure requirements	For the latest information, review the Azure network , storage , and compute requirements for Site Recovery. Azure Migrate has identical requirements for VMware migration.
Mobility service	The Mobility service agent must be installed on each VM you want to migrate.
UEFI boot	Supported. UEFI-based VMs will be migrated to Azure generation 2 VMs.
UEFI - Secure boot	Not supported for migration.

Support	Details
Target disk	VMs can only be migrated to managed disks (standard HDD, standard SSD, premium SSD) in Azure.
Disk size	up to 2-TB OS disk for gen 1 VM; up to 4-TB OS disk for gen 2 VM; 32 TB for data disks.
Disk limits	Up to 63 disks per VM.
Encrypted disks/volumes	VMs with encrypted disks/volumes aren't supported for migration.
Shared disk cluster	Not supported.
Independent disks	Supported.
Passthrough disks	Supported.
NFS	NFS volumes mounted as volumes on the VMs won't be replicated.
ReiserFS	Not supported.
iSCSI targets	Supported.
Multipath IO	Not supported.
Storage vMotion	Supported
Teamed NICs	Not supported.
IPv6	Not supported.
Guest/server disk with 4K logical and 4k physical sector size	Not supported.
Guest/server disk with 4K logical and 512-bytes physical sector size	Not supported.

Appliance requirements (agent-based)

The Azure Site Recovery Replication appliance is used to replicate machines to Azure in agent-based migration. [Learn more](#).

Set up the replication appliance using the OVA template for VMware agent-based migration. We recommend using this approach as it ensures all prerequisite configurations are handled by the template. The OVA template creates a machine with the required specifications. If your organization has restrictions, you can manually set up the replication appliance using PowerShell.

Ensure you meet all the [hardware](#) and [software](#) requirements, and any other prerequisites. [Learn more](#) on how to set up the appliance.

! Note

The replication appliance shouldn't be installed on a source machine that you want to replicate or on the Azure Migrate: Discovery and assessment appliance you might have installed before.

Port requirements (agent-based)

[Expand table](#)

Device	Connection
VMs	<p>The Mobility service running on VMs communicates with the on-premises replication appliance (configuration server) on port HTTPS 443 inbound, for replication management.</p> <p>VMs send replication data to the process server (running on the configuration server machine) on port HTTPS 9443 inbound. This port can be modified.</p>
Replication appliance	The replication appliance orchestrates replication with Azure over port HTTPS 443 outbound.
Process server	<p>The process server receives replication data, optimizes, and encrypts it, and sends it to Azure storage over port 443 outbound.</p> <p>By default the process server runs on the replication appliance.</p>

Azure VM requirements

All on-premises VMs replicated to Azure (with agentless or agent-based migration) must meet the Azure VM requirements summarized in this table.

[Expand table](#)

Component	Requirements
Guest operating system	<p>Verifies supported VMware VM operating systems for migration.</p> <p>You can migrate any workload running on a supported operating system.</p>
Guest operating system architecture	64-bit.

Component	Requirements
Operating system disk size	Up to 2,048 GB.
Operating system disk count	1
Data disk count	64 or less.
Data disk size	Up to 32 TB
Network adapters	Multiple adapters are supported.
Shared VHD	Not supported.
FC disk	Not supported.
BitLocker	<p>Not supported.</p> <p>BitLocker must be disabled before you migrate the machine.</p>
VM name	<p>From 1 to 63 characters.</p> <p>Restricted to letters, numbers, and hyphens.</p> <p>The machine name must start and end with a letter or number.</p>
Connect after migration-Windows	<p>To connect to Azure VMs running Windows after migration:</p> <ul style="list-style-type: none"> - Before migration, enable RDP on the on-premises VM. <p>Make sure that TCP and UDP rules are added for the Public profile, and that RDP is allowed in Windows Firewall > Allowed Apps for all profiles.</p> <p>For site-to-site VPN access, enable RDP and allow RDP in Windows Firewall > Allowed apps and features for Domain and Private networks.</p> <p>In addition, check that the operating system's SAN policy is set to OnlineAll. Learn more.</p>
Connect after migration-Linux	<p>To connect to Azure VMs after migration using SSH:</p> <p>Before migration, on the on-premises machine, check that the Secure Shell service is set to Start, and that firewall rules allow an SSH connection.</p> <p>After failover, on the Azure VM, allow incoming connections to the SSH port for the network security group rules on the failed over VM, and for the Azure subnet to which it's connected.</p> <p>In addition, add a public IP address for the VM.</p>

Next steps

- [Select](#) a VMware vSphere migration option.

Support matrix for Hyper-V migration

Article • 04/02/2025

This article summarizes support settings and limitations for migrating Hyper-V VMs with [Migration and modernization](#). If you're looking for information about assessing Hyper-V VMs for migration to Azure, review the [assessment support matrix](#).

✖ Caution

This article references CentOS, a Linux distribution that is End Of Life (EOL) status. Please consider your use and planning accordingly. For more information, see the [CentOS End Of Life guidance](#).

Migration limitations

You can select up to 10 VMs at once for replication. If you want to migrate more machines, replicate in groups of 10.

Hyper-V host requirements

expand Expand table

Support	Details
Deployment	The Hyper-V host can be standalone or deployed in a cluster. Azure Migrate replication software (Hyper-V Replication provider) is installed on the Hyper-V hosts.
Permissions	You need administrator permissions on the Hyper-V host.
Host operating system	Windows Server 2022, Windows Server 2019, Windows Server 2016, or Windows Server 2012 R2 with latest updates. Note that Server core installation of these operating systems is also supported.
Other Software requirements	.NET Framework 4.7 or later
Port access	Outbound connections on HTTPS port 443 to send VM replication data.

Hyper-V VMs

Support	Details
Operating system	All Windows and Linux operating systems that are supported by Azure.
Windows Server 2003	For VMs running Windows Server 2003, you need to install Hyper-V Integration Services before migration.
Linux VMs in Azure	<p>Some VMs might require changes so that they can run in Azure.</p> <p>For Linux, Azure Migrate makes the changes automatically for these operating systems:</p> <ul style="list-style-type: none"> - Red Hat Enterprise Linux 9.5, 9.x, 8.x, 7.9, 7.8, 7.7, 7.6, 7.5, 7.4, 7.0, 6.x - CentOS Stream - SUSE Linux Enterprise Server 15 SP4, 15 SP3, 15 SP2, 15 SP1, 15 SP0, 12, 11 SP4, 11 SP3 - Ubuntu 24.04, 22.04, 21.04, 20.04, 19.04, 19.10, 18.04LTS, 16.04LTS, 14.04LTS - Debian 11, 10, 9, 8, 7 - Oracle Linux 9, 8, 7.7-Cl, 7.7, 6 - Kali Linux (2016, 2017, 2018, 2019, 2020, 2021, 2022) - Alma Linux 8.x, 9.x - Rocky Linux 8.x, 9.x - For other operating systems, you make the required changes manually.
Required changes for Azure	Some VMs might require changes so that they can run in Azure. Make adjustments manually before migration. The relevant articles contain instructions about how to do this.
Linux boot	<p>If /boot is on a dedicated partition, it should reside on the OS disk, and not be spread across multiple disks.</p> <p>If /boot is part of the root (/) partition, then the '/' partition should be on the OS disk, and not span other disks.</p>
UEFI boot	Supported. UEFI-based VMs will be migrated to Azure generation 2 VMs.
UEFI - Secure boot	Not supported for migration.
Disk size	<p>Up to 2 TB OS disk for gen 1 VM; up to 4 TB OS disk for gen 2 VM; 32 TB for data disks.</p> <p>For existing Azure Migrate projects, you might need to upgrade the replication provider on the Hyper-V host to the latest version to replicate large disks up to 32 TB.</p>
Disk number	A maximum of 16 disks per VM.

Support	Details
Encrypted disks/volumes	Not supported for migration.
RDM/passthrough disks	Not supported for migration.
Shared disk	VMs using shared disks aren't supported for migration.
Ultra disk	Ultra disk migration isn't supported from the Azure Migrate portal. You have to do an out-of-band migration for the disks that are recommended as Ultra disks. That is, you can migrate selecting it as premium disk type and change it to Ultra disk after migration.
NFS	NFS volumes mounted as volumes on the VMs won't be replicated.
ReiserFS	Not supported.
iSCSI	VMs with iSCSI targets aren't supported for migration.
Target disk	You can migrate to Azure VMs with managed disks only.
IPv6	Not supported.
NIC teaming	Not supported.
Azure Site Recovery and/or Hyper-V	You can't replicate using Migration and modernization if the VM is enabled for replication with Azure Site Recovery or with Hyper-V replica.

URL access (public cloud)

The replication provider software on the Hyper-V hosts will need access to these URLs.

[\[+\] Expand table](#)

URL	Details
login.microsoftonline.com	Access control and identity management using Active Directory.
backup.windowsazure.com	Replication data transfer and coordination.
*.hypervrecoverymanager.windowsazure.com	Used for replication management.
*.blob.core.windows.net	Upload data to storage accounts.
dc.services.visualstudio.com	Upload app logs used for internal monitoring.

URL	Details
time.windows.com	Verifies time synchronization between system and global time.

URL access (Azure Government)

The replication provider software on the Hyper-V hosts will need access to these URLs.

[Expand table](#)

URL	Details
login.microsoftonline.us	Access control and identity management using Active Directory.
backup.windowsazure.us	Replication data transfer and coordination.
*.hypervrecoverymanager.windowsazure.us	Used for replication management.
*.blob.core.usgovcloudapi.net	Upload data to storage accounts.
dc.services.visualstudio.com	Upload app logs used for internal monitoring.
time.nist.gov	Verifies time synchronization between system and global time.

ⓘ Note

If your Migrate project has **private endpoint connectivity**, the replication provider software on the Hyper-V hosts will need access to these URLs for private link support.

- *.blob.core.windows.com - To access storage account that stores replicated data. This is optional and isn't required if the storage account has a private endpoint attached.
- login.windows.net for access control and identity management using Active Directory.

Replication storage account requirements

This table summarizes support for the replication storage account for Hyper-V VM migrations.

[\[+\] Expand table](#)

Setting	Support	Details
General purpose V2 storage accounts (Hot and Cool tier)	Supported	GPv2 storage accounts might incur higher transaction costs than V1 storage accounts.
Premium storage	Supported	However, standard storage accounts are recommended to help optimize costs. Cache storage account should be standard storage account and premium is not supported.
Region	Same region as virtual machine	Storage account should be in the same region as the virtual machine being protected.
Subscription	Can be different from source virtual machines	The Storage account need not be in the same subscription as the source virtual machine(s).
Azure Storage firewalls for virtual networks	Supported	If you're using firewall enabled replication storage account or target storage account, ensure you Allow trusted Microsoft services . Also, ensure that you allow access to at least one subnet of source virtual network. You should allow access from All networks for public endpoint connectivity.
Soft delete	Not supported	Soft delete isn't supported because once it's enabled on replication storage account, it increases cost. Azure Migrate performs very frequent creates/deletes of log files while replicating causing costs to increase.
Private endpoint	Supported	Follow the guidance to set up Azure Migrate with private endpoints .

Azure VM requirements

All on-premises VMs replicated to Azure must meet the Azure VM requirements summarized in this table.

[\[+\] Expand table](#)

Component	Requirements	Details
Operating system disk size	Up to 2,048 GB.	Check fails if unsupported.

Component	Requirements	Details
Operating system disk count	1	Check fails if unsupported.
Data disk count	16 or less.	Check fails if unsupported.
Data disk size	Up to 32 TB	Check fails if unsupported.
Network adapters	Multiple adapters are supported.	
Shared VHD	Not supported.	Check fails if unsupported.
FC disk	Not supported.	Check fails if unsupported.
BitLocker	Not supported.	BitLocker must be disabled before you enable replication for a machine.
VM name	<p>From 1 to 63 characters. Restricted to letters, numbers, and hyphens.</p> <p>The machine name must start and end with a letter or number.</p>	Update the value in the machine properties in Site Recovery.
Connect after migration-Windows	<p>To connect to Azure VMs running Windows after migration:</p> <ul style="list-style-type: none"> - Before migration, enable RDP on the on-premises VM. Make sure that TCP, and UDP rules are added for the Public profile, and that RDP is allowed in Windows Firewall > Allowed Apps, for all profiles. - For site-to-site VPN access, enable RDP and allow RDP in Windows Firewall -> Allowed apps and features for Domain and Private networks. In addition, check that the operating system's SAN policy is set to OnlineAll. Learn more. 	
Connect after migration-Linux	<p>To connect to Azure VMs after migration using SSH:</p> <ul style="list-style-type: none"> - Before migration, on the on-premises machine, check that the Secure Shell service is set to Start, and that firewall rules allow an SSH connection. 	

Component	Requirements	Details
	<ul style="list-style-type: none">- After migration, on the Azure VM, allow incoming connections to the SSH port for the network security group rules on the failed over VM, and for the Azure subnet to which it's connected. In addition, add a public IP address for the VM.	

Next steps

[Migrate Hyper-V VMs](#) for migration.

Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

Support matrix for migration of physical servers, AWS VMs, and GCP VMs

Article • 01/27/2025

This article summarizes support settings and limitations for migrating physical servers, Amazon Web Services (AWS) virtual machines (VMs), and Google Cloud Platform (GCP) VMs to Azure with [Migration and modernization](#). If you're looking for information about assessing physical servers for migration to Azure, see the [assessment support matrix](#).

Migrate machines as physical

You can migrate on-premises machines as physical servers by using agent-based replication. By using this tool, you can migrate a wide range of machines to Azure, such as:

- On-premises physical servers.
- VMs virtualized by platforms, such as Xen and KVM.
- Hyper-V VMs or VMware VMs, if for some reason you don't want to use the standard [Hyper-V](#) or [VMware](#) flows.
- VMs running in private clouds.
- VMs running in public clouds, including AWS or GCP.

Migration limitations

You can select up to 10 machines at once for replication. If you want to migrate more machines, replicate them in groups of 10.

Physical server requirements

The following table summarizes support for physical servers, AWS VMs, and GCP VMs that you want to migrate by using agent-based migration.

 [Expand table](#)

Support	Details
Machine workload	Azure Migrate and Modernize supports migration of any workload (such as Microsoft Entra ID or SQL Server) running on a supported machine.

Support	Details
Operating systems	For the latest information, see the operating system (OS) support for Azure Site Recovery. Azure Migrate and Modernize provides identical OS support.
Linux file system/guest storage	For the latest information, see the Linux file system support for Site Recovery. Azure Migrate and Modernize provides identical Linux file system support.
Network/Storage	For the latest information, see the network and storage prerequisites for Site Recovery. Azure Migrate and Modernize provides identical network/storage requirements.
Azure requirements	For the latest information, see the Azure network , storage , and compute requirements for Site Recovery. Azure Migrate and Modernize has identical requirements for physical server migration.
Mobility service	Install the Mobility service agent on each machine you want to migrate.
UEFI boot	Supported.
Windows : NTFS	Linux: The following filesystem types are supported: ext4, xfs, btrfs. Some filesystems such as ZFS, UFS, ReiserFS, and DazukoFS may not be supported subject to additional command requirements to mount them.
UEFI - Secure boot	Not supported for migration.
Target disk	Machines can be migrated only to managed disks (standard HDD, standard SSD, premium SSD) in Azure.
Ultra disk	Ultra disk migration isn't supported from the Azure Migrate and Modernize portal. You have to do an out-of-band migration for the disks that are recommended as Ultra disks. That is, you can migrate selecting it as premium disk type and change it to Ultra disk after migration.
Disk size	Up to 2-TB OS disk for gen 1 VM. Up to 4-TB OS disk for gen 2 VM and 32 TB for data disks.
Disk limits	Up to 63 disks per machine.
Encrypted disks/volumes	Machines with encrypted disks/volumes aren't supported for migration.
Shared disk cluster	Not supported.
Independent disks	Supported.
Passthrough disks	Supported.
NFS	NFS volumes mounted as volumes on the machines aren't replicated.

Support	Details
ReiserFS	Not supported.
iSCSI targets	Machines with iSCSI targets aren't supported for agentless migration.
Multipath IO	Supported for Windows servers with Microsoft or vendor-specific Device-Specific Module installed.
Teamed NICs	Not supported.
IPv6	Not supported.
PV drivers / XenServer tools	Not supported.

Replication appliance requirements

If you set up the replication appliance manually, make sure that it complies with the requirements summarized in the table. When you set up the Azure Migrate replication appliance as a VMware VM by using the Open Virtual Appliance template provided in the Azure Migrate and Modernize hub, the appliance is set up with Windows Server 2016 and complies with the support requirements.

- Learn about [replication appliance requirements](#).
- Install MySQL on the appliance. Learn about [installation options](#).
- Learn about [URLs](#) the replication appliance needs to access.

Azure VM requirements

All on-premises VMs replicated to Azure must meet the Azure VM requirements summarized in this table. When Site Recovery runs a prerequisites check for replication, the check fails if some of the requirements aren't met.

[\[+\] Expand table](#)

Component	Requirements	Details
Guest operating system	Verifies supported operating systems. You can migrate any workload running on a supported OS.	Check fails if unsupported.
Guest operating system architecture	64 bit.	Check fails if unsupported.

Component	Requirements	Details
Operating system disk size	Up to 2TB for Gen1 VM and 4TB for Gen2 VM.	Check fails if unsupported.
Operating system disk count	1.	Check fails if unsupported.
Data disk count	64 or less.	Check fails if unsupported.
Data disk size	Up to 32 TB.	Check fails if unsupported.
Network adapters	Multiple adapters are supported.	
Shared VHD	Not supported.	Check fails if unsupported.
FC disk	Not supported.	Check fails if unsupported.
BitLocker	Not supported.	Disable BitLocker before you enable replication for a machine.
VM name	<p>From 1 to 63 characters.</p> <p>Restricted to letters, numbers, and hyphens.</p> <p>The machine name must start and end with a letter or number.</p>	Update the value in the machine properties in Site Recovery.
Connect after migration-Windows	<p>To connect to Azure VMs running Windows after migration:</p> <ul style="list-style-type: none"> - Before migration enables Remote Desktop Protocol (RDP) on the on-premises VM. Make sure that TCP and UDP rules are added for the Public profile, and that RDP is allowed in Windows Firewall > Allowed Apps for all profiles. - For site-to-site virtual private network access, enable RDP and allow RDP in Windows Firewall > Allowed apps and features for Domain and Private networks. <p>Also check that the OS storage area network policy is set to OnlineAll. Learn more.</p>	
Connect after migration-Linux	<p>To connect to Azure VMs after migration by using Secure Shell (SSH):</p> <ul style="list-style-type: none"> - Before migration, on the on-premises machine, 	

Component	Requirements	Details
	<p>check that the SSH service is set to Start and that firewall rules allow an SSH connection.</p> <ul style="list-style-type: none">- After failover, on the Azure VM, allow incoming connections to the SSH port for the network security group rules on the failed over VM, and for the Azure subnet to which it's connected. Also add a public IP address for the VM.	

Next steps

[Migrate](#) physical servers.

Feedback

Was this page helpful?

 Yes

 No

[Provide product feedback](#) | [Get help at Microsoft Q&A](#)

Azure security baseline for Azure Migrate

Article • 02/25/2025

This security baseline applies guidance from the [Microsoft cloud security benchmark version 1.0](#) to Azure Migrate. The Microsoft cloud security benchmark provides recommendations on how you can secure your cloud solutions on Azure. The content is grouped by the security controls defined by the Microsoft cloud security benchmark and the related guidance applicable to Azure Migrate.

You can monitor this security baseline and its recommendations using Microsoft Defender for Cloud. Azure Policy definitions will be listed in the Regulatory Compliance section of the Microsoft Defender for Cloud portal page.

When a feature has relevant Azure Policy Definitions, they are listed in this baseline to help you measure compliance with the Microsoft cloud security benchmark controls and recommendations. Some recommendations may require a paid Microsoft Defender plan to enable certain security scenarios.

! Note

Features not applicable to Azure Migrate have been excluded. To see how Azure Migrate completely maps to the Microsoft cloud security benchmark, see the [full Azure Migrate security baseline mapping file](#).

Security profile

The security profile summarizes high-impact behaviors of Azure Migrate, which may result in increased security considerations.

Expand table

Service Behavior Attribute	Value
Product Category	Migration
Customer can access HOST / OS	No Access
Service can be deployed into customer's virtual network	True
Stores customer content at rest	True

Network security

For more information, see the [Microsoft cloud security benchmark: Network security](#).

NS-2: Secure cloud services with network controls

Features

Azure Private Link

Description: Service native IP filtering capability for filtering network traffic (not to be confused with NSG or Azure Firewall). [Learn more](#).

[] [Expand table](#)

Supported	Enabled By Default	Configuration Responsibility
True	False	Customer

Feature notes: The customer is able to configure Azure Migrate to require Private Link.

Configuration Guidance: Deploy private endpoints for all Azure resources that support the Private Link feature, to establish a private access point for the resources.

Azure Migrate Private Link support allows customers to securely discover, assess, and migrate servers over a private network while offering protection against data exfiltration risks and enabling greater migration velocity.

You can connect privately and securely to Azure Migrate over an Azure ExpressRoute private peering or a site-to-site (S2S) VPN connection by using Private Link.

Reference: [Using Azure Migrate with Azure Private Link](#)

Disable Public Network Access

Description: Service supports disabling public network access either through using service-level IP ACL filtering rule (not NSG or Azure Firewall) or using a 'Disable Public Network Access' toggle switch. [Learn more](#).

[] [Expand table](#)

Supported	Enabled By Default	Configuration Responsibility
True	False	Customer

Feature notes: The customer may configure Azure Migrate with Private Link, thereby disabling public endpoint access.

Configuration Guidance: Disable public network access using by toggling the switch for public network access.

You can configure the connectivity method for your Azure Migrate project and choose to enable or disable public network access to the Migrate project.

Reference: [Using Azure Migrate with private endpoints](#)

Identity management

For more information, see the [Microsoft cloud security benchmark: Identity management](#).

IM-1: Use centralized identity and authentication system

Features

Azure AD Authentication Required for Data Plane Access

Description: Service supports using Azure AD authentication for data plane access. [Learn more](#).

Expand table

Supported	Enabled By Default	Configuration Responsibility
True	True	Microsoft

Feature notes: Azure Migrate leverages managed identities and service principals in Azure AD for certain data plane operations. Azure Migrate also supports Azure AD for user access of the control plane through the Azure Portal.

Configuration Guidance: No additional configurations are required as this is enabled on a default deployment.

IM-3: Manage application identities securely and automatically

Features

Managed Identities

Description: Data plane actions support authentication using managed identities. [Learn more.](#)

[+] [Expand table](#)

Supported	Enabled By Default	Configuration Responsibility
True	False	Microsoft

Feature notes: Azure Migrate uses managed identity to securely access migrated data that is kept in a storage account. This is currently used in a scenario where private endpoint is configured.

Configuration Guidance: Use Azure managed identities instead of service principals when possible, which can authenticate to Azure services and resources that support Azure Active Directory (Azure AD) authentication. Managed identity credentials are fully managed, rotated, and protected by the platform, avoiding hard-coded credentials in source code or configuration files.

Service Principals

Description: Data plane supports authentication using service principals. [Learn more.](#)

[+] [Expand table](#)

Supported	Enabled By Default	Configuration Responsibility
True	False	Customer

Feature notes: The discovery and assessment services within Azure Migrate do not use service principals, however the migration service does use service principals in a public endpoint scenario to connect to the customer's key vault using Hyper-V recovery manager app.

Configuration Guidance: There is no current Microsoft guidance for this feature configuration. Please review and determine if your organization wants to configure this security feature.

IM-8: Restrict the exposure of credential and secrets

Features

Service Credential and Secrets Support Integration and Storage in Azure Key Vault

Description: Data plane supports native use of Azure Key Vault for credential and secrets store. [Learn more.](#)

[] [Expand table](#)

Supported	Enabled By Default	Configuration Responsibility
True	True	Microsoft

Feature notes: The Azure Migrate appliance leverages Key Vault to manage connection strings for the service bus, and access keys for the storage accounts are used for replication.

Configuration Guidance: No additional configurations are required as this is enabled on a default deployment.

Data protection

For more information, see the [Microsoft cloud security benchmark: Data protection](#).

DP-3: Encrypt sensitive data in transit

Features

Data in Transit Encryption

Description: Service supports data in-transit encryption for data plane. [Learn more.](#)

[] [Expand table](#)

Supported	Enabled By Default	Configuration Responsibility
True	True	Microsoft

Feature notes: Encryption of data-in-transit is enabled by default. Azure Migrate supports data encryption in transit with TLS v1.2 or greater.

While this is optional for traffic on private networks, this is critical for traffic on external and public networks. For HTTP traffic, ensure that any clients (including the Azure Migrate appliance and other machines on which you've installed Azure Migrate software) connecting to

your Azure resources can negotiate TLS v1.2 or greater. For remote management, use SSH (for Linux) or RDP/TLS (for Windows) instead of an unencrypted protocol. Obsolete SSL, TLS, and SSH versions and protocols, and weak ciphers should be disabled.

Configuration Guidance: No additional configurations are required as this is enabled on a default deployment.

DP-4: Enable data at rest encryption by default

Features

Data at Rest Encryption Using Platform Keys

Description: Data at-rest encryption using platform keys is supported, any customer content at rest is encrypted with these Microsoft managed keys. [Learn more](#).

 Expand table

Supported	Enabled By Default	Configuration Responsibility
True	True	Microsoft

Feature notes: All data persisted in Azure Migrate is encrypted at rest with Microsoft-managed keys.

The Server Migration tool in Azure Migrate replicates data from the disks of servers being migrated to storage accounts and managed disks in your Azure subscription. Data handling is transient until it is written to storage accounts or managed disks in the subscription and is not persisted in Azure Migrate. Replicated data on the storage account and managed disks is encrypted at rest with Microsoft-managed keys. For highly sensitive data, you have options to implement additional encryption at rest with customer-managed keys on the storage account and managed disks.

Configuration Guidance: No additional configurations are required as this is enabled on a default deployment.

DP-5: Use customer-managed key option in data at rest encryption when required

Features

Data at Rest Encryption Using CMK

Description: Data at-rest encryption using customer-managed keys is supported for customer content stored by the service. [Learn more](#).

[+] [Expand table](#)

Supported	Enabled By Default	Configuration Responsibility
True	False	Customer

Configuration Guidance: Enable and implement data at rest encryption for your replicated data using customer-managed keys.

To replicate VMs with CMK, you'll need to create a disk encryption set under the target Resource Group. A disk encryption set object maps Managed Disks to a Key Vault that contains the CMK to use for SSE.

Reference: [Migrate VMware VMs to Azure \(agentless\)](#)

DP-6: Use a secure key management process

Features

Key Management in Azure Key Vault

Description: The service supports Azure Key Vault integration for any customer keys, secrets, or certificates. [Learn more](#).

[+] [Expand table](#)

Supported	Enabled By Default	Configuration Responsibility
True	True	Microsoft

Feature notes: The Azure Migrate appliance leverages Key Vault to manage connection strings for the service bus, and access keys for the storage accounts are used for replication.

Configuration Guidance: No additional configurations are required as this is enabled on a default deployment.

Reference: [Set up the Azure Migrate appliance](#)

Asset management

For more information, see the [Microsoft cloud security benchmark: Asset management](#).

AM-2: Use only approved services

Features

Azure Policy Support

Description: Service configurations can be monitored and enforced via Azure Policy. [Learn more](#).

Expand table

Supported	Enabled By Default	Configuration Responsibility
True	False	Customer

Feature notes: Azure Migrate may leverage Azure Policy, however it must be configured by the customer to be enforced.

Configuration Guidance: Use Microsoft Defender for Cloud to configure Azure Policy to audit and enforce configurations of your Azure resources. Use Azure Monitor to create alerts when there is a configuration deviation detected on the resources. Use Azure Policy [deny] and [deploy if not exists] effects to enforce secure configuration across Azure resources.

Reference: [Azure Policy built-in definitions for Azure Migrate](#)

Logging and threat detection

For more information, see the [Microsoft cloud security benchmark: Logging and threat detection](#).

LT-4: Enable logging for security investigation

Features

Azure Resource Logs

Description: Service produces resource logs that can provide enhanced service-specific metrics and logging. The customer can configure these resource logs and send them to their own data sink like a storage account or log analytics workspace. [Learn more](#).

 [Expand table](#)

Supported	Enabled By Default	Configuration Responsibility
False	Not Applicable	Not Applicable

Configuration Guidance: This feature is not supported to secure this service.

Next steps

- See the [Microsoft cloud security benchmark overview](#)
- Learn more about [Azure security baselines](#)

How does Hyper-V replication work?

11/29/2024

This article provides an overview of the architecture and processes used when you migrate Hyper-V VMs with the Migration and modernization tool.

[Azure Migrate](#) provides a central hub to track discovery, assessment, and migration of your on-premises apps and workloads, and private/public cloud VMs, to Azure. The hub provides Azure Migrate tools for assessment and migration, as well as third-party independent software vendor (ISV) offerings.

Agentless migration

The Migration and modernization tool provides agentless replication for on-premises Hyper-V VMs, using a migration workflow that's optimized for Hyper-V. You install a software agent only on Hyper-V hosts or cluster nodes. Nothing needs to be installed on Hyper-V VMs.

Migration and modernization and Azure Site Recovery

Migration and modernization is a tool for migrating on-premises workloads, and cloud-based VMs, to Azure. Site Recovery is a disaster recovery tool. The tools share some common technology components used for data replication, but serve different purposes.

Architectural components



Component	Deployment
Replication provider	The Microsoft Azure Site Recovery provider is installed on Hyper-V hosts, and registered with the Migration and modernization tool. The provider orchestrates replication for Hyper-V VMs.
Recovery Services agent	The Microsoft Azure Recovery Service agent handles data replication. It works with the provider to replicate data from Hyper-V VMs to Azure. The replicated data is uploaded to a storage account in your Azure subscription. The Migration and modernization tool processes the replicated data, and applies it to replica disks in the subscription. The replica disks are used to create the Azure VMs when you migrate.

- Components are installed by a single setup file, downloaded from the Migration and modernization tool in the portal.
- The provider and appliance use outbound HTTPS port 443 connections to communicate with the Migration and modernization tool.
- Communications from the provider and agent are secure and encrypted.

Replication process

1. When you enable replication for a Hyper-V VM, initial replication begins.
2. A Hyper-V VM snapshot is taken.
3. VHDs on the VM are replicated one-by-one, until they're all copied to Azure. Initial replication time depends on the VM size, and network bandwidth.
4. Disk changes that occur during initial replication are tracked using Hyper-V Replica, and stored in log files (hrl files).
 - Log files are in the same folder as the disks.
 - Each disk has an associated hrl file that's sent to secondary storage.
 - The snapshot and log files consume disk resources while initial replication is in progress.
5. After initial replication finishes, the VM snapshot is deleted, and delta replication begins.
6. Incremental disk changes are tracked in hrl files. Replication logs are periodically uploaded to an Azure storage account by the Recovery Services agent.

Performance and scaling

Replication performance for Hyper-V is influenced by factors that include VM size, the data change rate (churn) of the VMs, available space on the Hyper-V host for log file storage,

upload bandwidth for replication data, and target storage in Azure.

- If you're replicating multiple machines at the same time, use the [Azure Site Recovery Deployment Planner](#) for Hyper-V, to help optimize replication.
- Plan your Hyper-V replication, and distribute replication over Azure storage accounts, in accordance with capacity.

Control upload throughput

You can limit the amount of bandwidth used to upload data to Azure on each Hyper-V host. Be careful. If you set the values too low it will adversely impact replication, and delay migration.

1. Sign in to the Hyper-V host or cluster node.
2. Run **C:\Program Files\Microsoft Azure Recovery Services Agent\bin\wabadmin.msc**, to open the Windows Azure Backup MMC snap-in.
3. In the snap-in, select **Change Properties**.
4. In **Throttling**, select **Enable internet bandwidth usage throttling for backup operations**. Set the limits for work and non-work hours. Valid ranges are from 512 Kbps to 1,023 Mbps.

Influence upload efficiency

If you have spare bandwidth for replication, and want to increase uploads, you can increase the number of threads allocated for the upload task, as follows:

1. Open the registry with Regedit.
2. Navigate to key **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Azure Backup\Replication\UploadThreadsPerVM**
3. Increase the value for the number of threads used for data upload for each replicating VM. The default value is 4 and the max value is 32.

Next steps

Try out [Hyper-V migration](#) using the Migration and modernization tool.

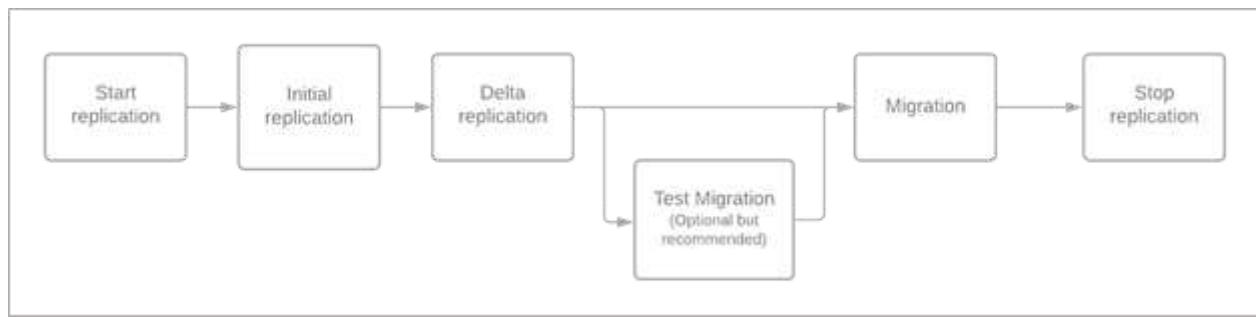
Agentless migration architecture

08/08/2025

This article describes the replication concepts when you're migrating VMware virtual machines (VMs) by using the agentless migration method in Azure Migrate.

Replication process

The agentless replication option works by using VMware snapshots and VMware changed block tracking (CBT) technology to replicate data from VM disks. The following block diagram shows you the steps involved when you migrate your VMs by using Azure Migrate.



When you configure replication for a VM, it goes through an initial replication phase. During this phase, Azure Migrate takes a VM snapshot. The service then replicates a full copy of data from the snapshot disks to managed disks in your target subscription.

After initial replication for the VM is complete, the replication process transitions to an incremental replication (delta replication) phase. In this phase, data changes that occurred since the beginning of the last completed replication cycle are replicated and written to the replica managed disks. This part of the process keeps replication in sync with changes on the VM.

Azure Migrate uses VMware CBT technology to keep track of changes between replication cycles. At the start of a replication cycle, Azure Migrate takes a VM snapshot. The service uses CBT to get the changes between the current snapshot and the last successfully replicated snapshot. Only the data that changed since the previous completed replication cycle is replicated, to keep replication for the VM in sync. At the end of each replication cycle, the snapshot is released, and Azure Migrate performs snapshot consolidation for the VM.

When you perform the migration operation on a replicating VM, an on-demand delta replication cycle replicates the remaining changes since the last replication cycle. After the on-demand cycle finishes, Azure Migrate creates the VM in Azure by using the replica managed disks that correspond to the VM.

Before you trigger the migration, you must shut down the on-premises VM. Shutting down the VM prevents data loss during migration.

After the migration is successful and the VM restarts in Azure, ensure that you stop the replication of the VM. Stopping the replication deletes the intermediate disks (seed disks) that were created during data replication. You then avoid incurring extra charges associated with the storage transactions on these disks.

Replication cycles

! Note

Be sure to check for any existing snapshots on the VM from earlier replication attempts, partner apps, or active backup tools (e.g., VEEAM), as this will block agentless replication setup in Azure Migrate. Snapshot-based backups conflict with Azure Migrate's agentless change tracking and replication process and should not be used concurrently.

A replication cycle is the periodic process of transferring data from an on-premises environment to Azure managed disks. A full replication cycle consists of the following steps:

1. Create a VMware snapshot for each disk associated with the VM.
2. Upload data to a log storage account in Azure.
3. Release the snapshot.
4. Consolidate VMware disks.

A cycle is complete after the disks are consolidated.

Components for replication

An Azure Migrate appliance has the following *on-premises* components that are responsible for replication:

- Data replication agent
- Gateway agent

The following table summarizes the *Azure* components that are created when you use the agentless method of VMware VM migration.

 Expand table

Component	Region	Subscription	Description
Recovery services vault	Azure Migrate project's region	Azure Migrate project's subscription	Vault that's used to orchestrate data replication.
Service bus	Target region	Azure Migrate project's subscription	Component that's used for communication between the cloud service and the Azure Migrate appliance.
Log storage account	Target region	Azure Migrate project's subscription	Account that's used to store replication data. The service reads this data and applies it on the customer's managed disk.
Gateway storage account	Target region	Azure Migrate project's subscription	Account that's used to store machine states during replication
Key vault	Target region	Azure Migrate project's subscription	Vault that manages connection strings for the service bus and access keys for the log storage account.
Virtual machine	Target region	Target subscription	VM created in Azure when you migrate.
Managed disks	Target region	Target subscription	Managed disks attached to Azure VMs.
Network interface cards (NICs)	Target region	Target subscription	NICs attached to the VMs created in Azure.

Required permissions

When you start replication for the first time, the logged-in user must have the following roles:

- Owner or Contributor and User Access Administrator on the Azure Migrate project's resource group and the target resource group

For the subsequent replications, the logged-in user must have the following roles:

- Owner or Contributor on the Azure Migrate project's resource group and the target resource group

In addition to the preceding roles, the logged-in user needs the following permission at a subscription level: `Microsoft.Resources/subscriptions/resourceGroups/read`.

Data integrity

There are two stages in every replication cycle, to help ensure data integrity between the on-premises disk (source disk) and the replica disk in Azure (target disk).

Validate replication

The first stage validates that every sector that changed in the source disk is replicated to the target disk. You perform the validation by using bitmaps.

The source disk is divided into sectors of 512 bytes. Every sector in the source disk is mapped to a bit in the bitmap. When data replication starts, Azure Migrate creates a bitmap for all the changed blocks (in delta cycle) in the source disk that needs to be replicated. Similarly, when the data is transferred to the target Azure disk, Azure Migrate creates a bitmap.

After the data transfer finishes successfully, the cloud service compares the two bitmaps to ensure that it didn't miss any changed block. If there's any mismatch between the bitmaps, the cycle is considered failed. Because every cycle is resynchronization, the mismatch is fixed in the next cycle.

Check replicated data

The second stage ensures that the data that's transferred to the Azure disks is the same as the data that was replicated from the source disks.

Every changed block that's uploaded is compressed and encrypted before it's written as a blob in the log storage account. Azure Migrate computes the checksum of this block before compression. This checksum is stored as metadata along with the compressed data.

Upon decompression, Azure Migrate calculates the checksum for the data and compares it with the checksum computed in the source environment. If there's a mismatch, the data isn't written to the Azure disks, and the cycle is considered failed. Because every cycle is resynchronization, the mismatch is fixed in the next cycle.

Security

The Azure Migrate appliance compresses data and encrypts it before uploading it. Data is transmitted over a secure communication channel that uses HTTPS and TLS 1.2 or later. Additionally, Azure Storage automatically encrypts your data when it's persisted to the cloud (encryption at rest).

Replication status

When a VM undergoes replication (data copy), there are several possible states:

- **Initial replication queued:** The VM is queued for replication or migration because other VMs might be consuming the on-premises resources during replication or migration. After the resources are free, this VM is processed.
- **Initial replication in progress:** The VM is scheduled for initial replication.
- **Initial replication:** The VM is undergoing initial replication. When the VM is undergoing initial replication, you can't proceed with test migration and production migration. You can only stop replication at this stage.
- **Initial replication (x%):** The initial replication is active and has progressed by the shown percentage.
- **Delta sync:** The VM might be undergoing a delta replication cycle that replicates the remaining data churn since the last replication cycle.
- **Pause in progress:** The VM is undergoing an active delta replication cycle and is paused.
- **Paused:** The replication cycles are paused. You can resume the replication cycles by performing the operation to resume replication.
- **Resume queued:** The VM is queued for resuming replication because other VMs are currently consuming the on-premises resources.
- **Resume in progress (x%):** The replication cycle is being resumed for the VM and has progressed by the shown percentage.
- **Stop replication in progress:** Replication cleanup is in progress. When you stop replication, the intermediate managed disks (seed disks) created during replication are deleted. You can learn more about stopping replication [later in this article](#).
- **Complete migration in progress:** Migration cleanup is in progress. When you complete migration, the intermediate managed disks (seed disks) created during replication are deleted. You can learn more about completing replication [later in this article](#).
- **– :** When the VM is successfully migrated or when you stop replication, the status changes to a dash. After you complete migration or stop replication and the operation finishes successfully, the VM is removed from the list of replicating machines. You can find the VM on the tab for virtual machines in the replication wizard.

Other states

- **Initial replication failed:** The initial data couldn't be copied for the VM. Follow the remediation guidance to resolve.
- **Repair pending:** There was a problem in the replication cycle. You can select the link to understand possible causes and actions to remediate (as applicable). If you opted for **Automatically repair replication** by selecting **Yes** when you triggered replication of VM, the tool tries to repair it for you. Otherwise, select the VM, and then select **Repair Replication**.

If you didn't opt for **Automatically repair replication** or if the repair step didn't work for you, stop replication for the VM. Reset the CBT on the VM, and then reconfigure the replication.

- **Repair replication queued:** The VM is queued for replication repair because other VMs are consuming the on-premises resources. After the resources are free, the VM is processed for repair replication.
- **Resync (x%):** The VM is undergoing a data resynchronization. This resynchronization can happen if there was a problem or a mismatch during data replication.
- **Stop replication/complete migration failed:** Select the link to understand the possible causes for failure and actions to remediate (as applicable).

! **Note**

Some VMs go into a queued state to ensure minimal impact on the source environment due to the consumption of storage input/output operations per second (IOPS). These VMs are processed based on the scheduling logic, as described [later in this article](#).

Status of the test migration or production migration

- **Test migration pending:** The VM is in the delta replication phase. You can now perform test migration (or production migration).
- **Test migration clean up pending:** After test migration is complete, perform a test migration cleanup to avoid charges in Azure.
- **Ready to migrate:** The VM is ready for migration to Azure.
- **Migration in progress queued:** The VM is queued for migration because other VMs are consuming the on-premises resources during replication (or migration). After the resources are free, the VM is processed.
- **Test migration/Migration in progress:** The VM is undergoing a test migration or a production migration. You can select the link to check the ongoing migration job.
- **Date, timestamp:** The test migration or production migration happened at this date and time.
- **-:** Initial replication is in progress. You can perform a test migration or a production migration after the replication process transitions to a delta sync (incremental replication) phase.

Other states

- **Completed with info:** The test migration or production migration job finished with information. You can select the link to check the last migration job for possible causes and actions to remediate (as applicable).
- **Failed:** The test migration or production migration job failed. You can select the link to check the last migration job for possible causes and actions to remediate.

Scheduling logic

Initial replication is scheduled when you configure replication for a VM. Incremental replications (delta replications) follow it.

Delta replication cycles are scheduled as follows:

- First delta replication cycle is scheduled immediately after the initial replication cycle finishes.
- Next delta replication cycles are scheduled according to the following logic: `min[max[1 hour, (<Previous delta replication cycle time>/2)], 12 hours]`.

That is, the next delta replication is scheduled no sooner than 1 hour and no later than 12 hours. For example, if a VM takes 4 hours for a delta replication cycle, the next delta replication cycle is scheduled in 2 hours, and not in the next hour.

ⓘ Note

The scheduling logic is different after the initial replication finishes. The first delta cycle is scheduled immediately after the initial replication finishes. Subsequent cycles follow the scheduling logic.

- When you trigger migration, an on-demand (pre-failover) delta replication cycle occurs for the VM before migration.

Prioritization

Here's the prioritization of VMs for various stages of replication:

- Ongoing VM replications are prioritized over scheduled replications (new replications).
- The on-demand (pre-failover) delta replication cycle has the highest priority, followed by the initial replication cycle. The delta replication cycle has the lowest priority.

Whenever you trigger a migration operation, the on-demand replication cycle for the VM is scheduled. Other ongoing replications have to wait if they're competing for resources.

Constraints

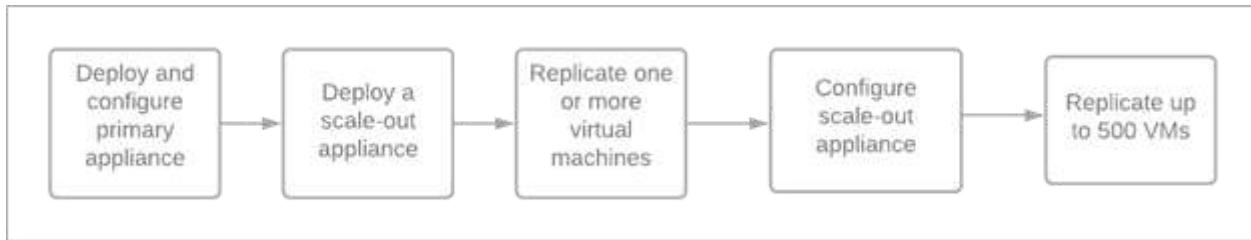
The following constraints help ensure that you don't exceed the IOPS limits on the storage area networks:

- Each Azure Migrate appliance supports the replication of 52 disks in parallel.
- Each ESXi host supports 8 disks. Every ESXi host has a 32-MB NFC buffer. So, you can schedule 8 disks on the host. (Each disk takes up 4 MB of buffer for incident response and disaster recovery.)
- Each datastore can have a maximum of 15 disk snapshots. The only exception is when more than 15 disks are attached to a VM.

Scale-out replication

Azure Migrate supports concurrent replication of 500 VMs. When you plan to replicate more than 300 VMs, you must deploy a scale-out appliance. The scale-out appliance is similar to an Azure Migrate primary appliance but consists only of a gateway agent to facilitate data transfer to Azure.

The following diagram shows the recommended way to use the scale-out appliance.



You can deploy the scale-out appliance anytime after you configure the primary appliance, but it isn't required until 300 VMs are replicating concurrently. When 300 VMs are replicating concurrently, you must deploy the scale-out appliance to proceed.

Stopping replication or completing a migration

When you stop replication, the intermediate managed disks (seed disks) created during replication are deleted. You can stop replication only during an active replication. You can select **Complete migration** to stop the replication after the VM is migrated.

You can replicate the VM for which the replication is stopped by enabling replication again. If the VM was migrated, you can resume replication and migration again.

As a best practice, you should always complete the migration after the VM migrates successfully to Azure. This practice ensures that you don't incur extra charges for storage

transactions on the intermediate managed disks (seed disks).

In some cases, stopping replication takes time. The reason is that whenever you stop replication, the ongoing replication cycle is completed (only when the VM is in delta sync) before deletion of the artifacts.

Impact of churn

You can minimize the amount of data transfer in each replication cycle by allowing the data to fold as much as possible before you schedule the next cycle. Because agentless replication folds in data, the *churn pattern* is more important than the *churn rate*. When a file is written again and again, the rate doesn't have much impact. However, a pattern in which every other sector is written causes high churn in the next cycle.

If the current delta replication cycle experiences delays due to high data churn, the initiation of the subsequent cycle might be delayed. A higher volume of data to replicate for a specific disk extends the duration required to create a recovery point. As a result, the final migration cycle takes longer. This situation leads to an extended shutdown window for the source VM.

If the snapshot size increases (due to churn pattern) to an extent that crosses the available capacity of the datastore, there's a risk of the datastore running out of space. This situation can adversely affect production workloads and might render the source VM unresponsive.

To mitigate this risk, we recommend that you increase the datastore size proactively. If multiple VMs that you're replicating concurrently have disks in a datastore that has low available capacity, we advise that you perform migrations one VM at a time to avoid resource contention.

Management of replication

Throttling

You can increase or decrease the replication bandwidth by using `NetQosPolicy`. The `AppNamePrefix` value to use in `NetQosPolicy` is `GatewayWindowsService.exe`.

To throttle replication traffic from the Azure Migrate appliance, you can create a policy like the following example on the appliance. This policy applies to all the replicating VMs from the Azure Migrate appliance simultaneously.

```
New-NetQosPolicy -Name "ThrottleReplication" -AppPathNameMatchCondition  
"GatewayWindowsService.exe" -ThrottleRateActionBitsPerSecond 1MB
```

You can also increase and decrease replication bandwidth based on a schedule by using the [sample script](#).

Blackout window

Azure Migrate provides a configuration-based mechanism that you can use to specify the time interval during which you don't want any replications to proceed. This interval is called the *blackout window*. The need for a blackout window can arise in multiple scenarios, such as when the source environment is resource constrained or when you want replication to happen only outside business hours.

Note

The existing replication cycles at the start of the blackout window finish before the replication pauses.

For any migration that you initiate during the blackout window, the final replication doesn't run. The migration fails.

You can specify a blackout window for the appliance by creating or updating the `GatewayDataWorker.json` file in `C:\ProgramData\Microsoft Azure\Config`. A typical file has this form:

```
{  
  "BlackoutWindows": "List of blackout windows"  
}
```

The list of blackout windows is a pipe-delimited (|) string of the format `<DayOfWeek>;<StartTime>;<Duration>`. You can specify the duration in days, hours, and minutes. For example, you can specify the blackout windows as:

```
{  
  "BlackoutWindows": "Monday;7:00;7h | Tuesday;8:00;1d7h | Wednesday;16:00;1d |  
  Thursday;18:00;5h | Friday;13:00;8m"  
}
```

The first value in the preceding example indicates a blackout window that starts every Monday at 7:00 AM local time (time on the appliance) and lasts 7 hours.

After you create or update `GatewayDataWorker.json` with these contents, you need to restart the gateway service on the appliance for these changes to take effect.

In the scale-out scenario, the primary appliance and the scale-out appliance honor the blackout windows independently. As a best practice, we recommend keeping the windows consistent across appliances.

Related content

- [Migrate VMware VMs to Azure \(agentless\)](#)

Prepare for VMware agentless migration

05/13/2025

This article provides an overview of the changes performed when you [migrate VMware VMs to Azure via the agentless migration](#) method using the Migration and modernization tool.

Caution

This article references CentOS, a Linux distribution that is End Of Life (EOL) status. Please consider your use and planning accordingly. For more information, see the [CentOS End Of Life guidance](#).

Before you migrate your on-premises VM to Azure, you may require a few changes to make the VM ready for Azure. These changes are important to ensure that the migrated VM can boot successfully in Azure and connectivity to the Azure VM can be established. Azure Migrate automatically handles these configuration changes for the following operating system versions for both Linux and Windows. This process is called *Hydration*.

Note

If a major version of an operating system is supported in agentless migration, all minor versions and kernels are automatically supported.

Operating system versions supported for hydration

- Windows Server 2008 or later
- Red Hat Enterprise Linux 9.5, 9.x, 8.x, 7.9, 7.8, 7.7, 7.6, 7.5, 7.4, 7.3, 7.2, 7.1, 7.0, 6.x
- CentOS Stream
- SUSE Linux Enterprise Server 15 SP6, 15 SP5, 15 SP4, 15 SP3, 15 SP2, 15 SP1, 15 SP0, 12, 11 SP4, 11 SP3
- Ubuntu 22.04, 21.04, 20.04, 19.04, 19.10, 18.04LTS, 16.04LTS, 14.04LTS
- Kali Linux (2016, 2017, 2018, 2019, 2020, 2021, 2022)
- Debian 11, 10, 9, 8, 7
- Oracle Linux 9, 8, 7.7-Cl, 7.7, 6
- Alma Linux 8.x, 9.x
- Rocky Linux 8.x, 9.x

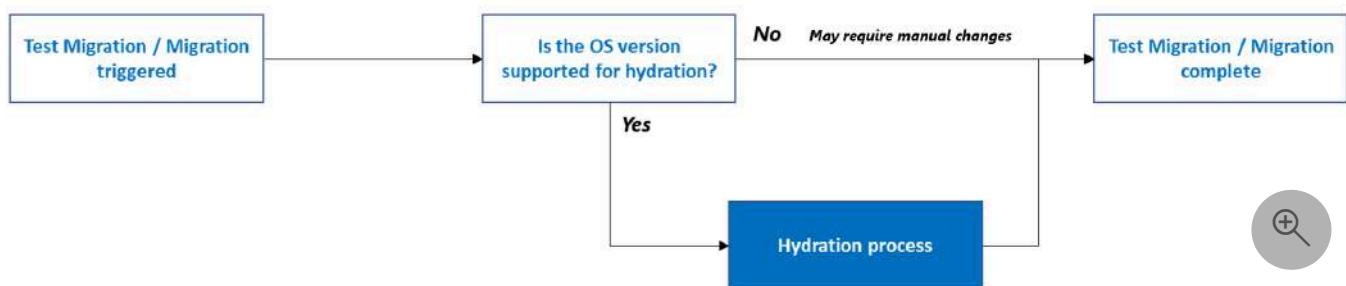
You can also use this article to manually prepare the VMs for migration to Azure for operating systems versions not listed above. At a high level, these changes include:

- Validate the presence of the required drivers

- Enable the serial console
- Configure network settings
- Install the VM guest agent

Hydration process

You have to make some changes to the VMs configuration before the migration to ensure that the migrated VMs function properly on Azure. Azure Migrate handles these configuration changes via the *hydration* process. The hydration process is only performed for the versions of Azure supported operating systems given above. Before you migrate, you may need to perform the required changes manually for other operating system versions that aren't listed above. If the VM is migrated without the required changes, the VM may not boot, or you may not have connectivity to the migrated VM. The following diagram shows you that Azure Migrate performs the hydration process.

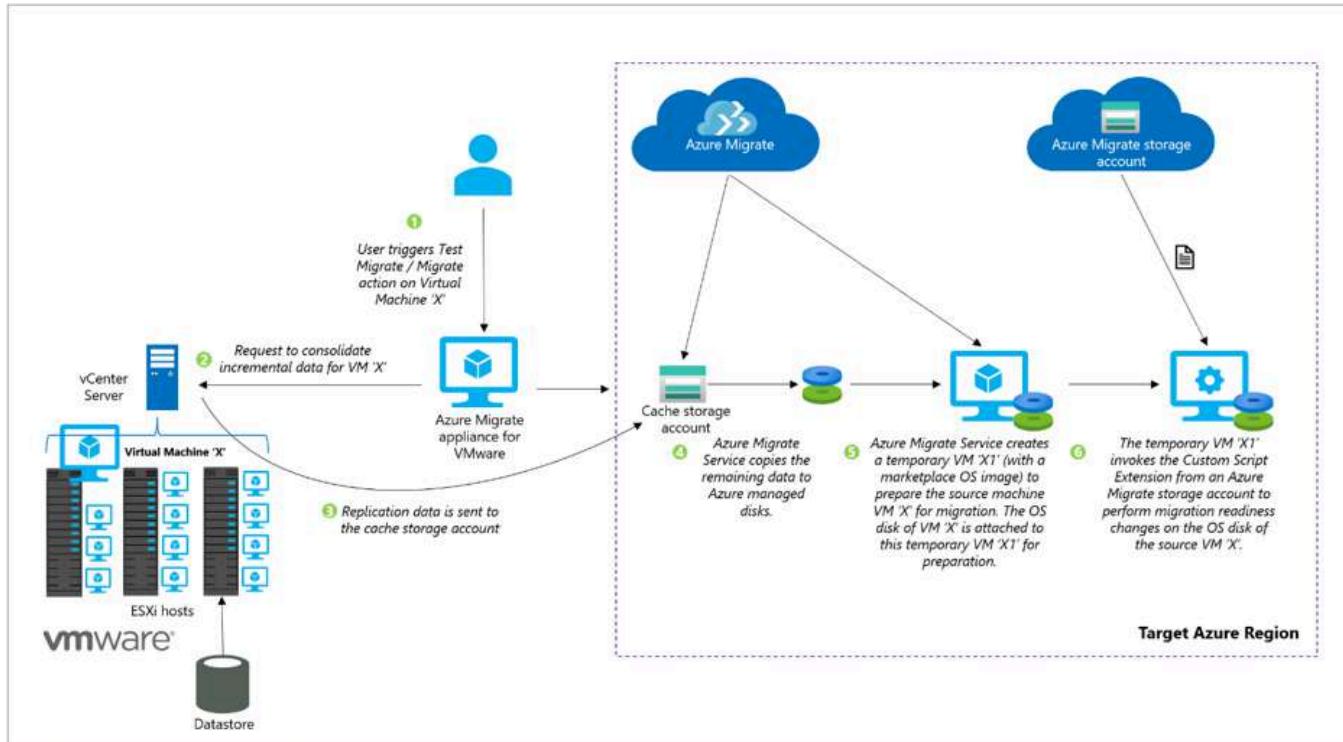


When a user triggers *Test Migrate* or *Migrate*, Azure Migrate performs the hydration process to prepare the on-premises VM for migration to Azure. To set up the hydration process, Azure Migrate creates a temporary Azure VM and attaches the disks of the source VM to perform changes to make the source VM ready for Azure. The temporary Azure VM is an intermediate VM created during the migration process before the final migrated VM is created. The temporary VM will be created with a similar OS type (Windows/Linux) using one of the marketplace OS images. If the on-premises VM is running Windows, the operating system disk of the on-premises VM will be attached as a data disk to the temporary VM for performing changes. If it's a Linux server, all the disks attached to the on-premises VM will be attached as data disks to the temporary Azure VM.

Azure Migrate will create the network interface, a new virtual network, subnet, and a network security group (NSG) to host the temporary VM. These resources are created in the customer's subscription. If there are conflicting policies that prevent the creation of the network artifacts, Azure Migrate will attempt to create the temporary Azure VM in the virtual network and subnet provided as part of the replication target settings options.

After the virtual machine is created, Azure Migrate will invoke the [Custom Script Extension](#) on the temporary VM using the Azure Virtual Machine REST API. The Custom Script Extension

utility will execute a preparation script containing the required configuration for Azure readiness on the on-premises VM disks attached to the temporary Azure VM. The preparation script is downloaded from an Azure Migrate owned storage account. The network security group rules of the virtual network will be configured to permit the temporary Azure VM to access the Azure Migrate storage account for invoking the script.



! Note

Hydration VM disks do not support Customer Managed Key (CMK). Platform Managed Key (PMK) is the default option.

Changes performed during the hydration process

The preparation script executes the following changes based on the OS type of the source VM to be migrated. You can also use this section as a guide to manually prepare the VMs for migration for operating systems versions not supported for hydration.

Changes performed on Windows servers

1. Discover and prepare the Windows OS volume

Before performing relevant configuration changes, the preparation script will validate if the correct OS disk was selected for migration. The preparation script will look through all

the attached volumes visible to the system and look for the SYSTEM registry hive file path to find the source OS volume.

The following actions are performed in this step:

- Mounts each partition on the OS disk attached to the temporary VM.
- Looks for \Windows\System32\Config\System registry files after mounting the partition.
- If the files aren't found, the partition is unmounted, and the search continues for the correct partition.
- If the files aren't present on any of the partitions, it could indicate that an incorrect OS disk was selected, or the OS disk is corrupted. Azure Migrate will fail the migration process with an appropriate error.

 **Note**

This step isn't relevant if you're manually preparing the servers for migration.

2. Make boot and connectivity related changes

After the source OS volume files are detected, the preparation script will load the SYSTEM registry hive into the registry editor of the temporary Azure VM and perform the following changes to ensure VM boot up and connectivity. You need to configure these settings manually if the OS version isn't supported for hydration.

a. Validate the presence of the required drivers

Ensure if the required drivers are installed and are set to load at **boot start**. These Windows drivers allow the server to communicate with the hardware and other connected devices.

- IntelIde.sys
- Atapi
- Storflt
- Storvsc
- VMbus

b. Set storage area network (SAN) policy to Online All

This ensures that the Windows volumes in the Azure VM use the same drive letter assignments as the on-premises VM. By default, Azure VMs are assigned drive D: to

use as temporary storage. This drive assignment causes all other attached storage drive assignments to increment by one letter. To prevent this automatic assignment, and to ensure that Azure assigns the next free drive letter to its temporary volume, set the storage area network (SAN) policy to Online All.

To manually configure this setting:

- On the on-premises server, open the command prompt with elevated privileges and enter `diskpart`.

```
Administrator: Command Prompt - diskpart
Microsoft Windows [Version 10.0.19042.867]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>diskpart

Microsoft DiskPart version 10.0.19041.610

Copyright (C) Microsoft Corporation.
On computer: MININT-B19D4SM

DISKPART>
```

- Enter SAN. If the drive letter of the guest operating system isn't maintained, Offline All or Offline Shared is returned.
- At the DISKPART prompt, enter `SAN Policy=OnlineAll`. This setting ensures that disks are brought online, and that you can read and write to both disks.

```
DISKPART> SAN Policy = onlineall
diskPart successfully changed the SAN policy for the current operating system.

DISKPART> SAN
SAN Policy : Online All

DISKPART> exit
Leaving DiskPart...

C:\WINDOWS\system32>
```

3. Set the DHCP start type

The preparation script will also set the DHCP service start type as Automatic. This will enable the migrated VM to obtain an IP address and establish connectivity post-migration. Make sure the DHCP service is configured, and the status is running.

PS C:\WINDOWS\system32> Get-Service -Name Dhcp		
Status	Name	DisplayName
Running	Dhcp	DHCP Client

To edit the DHCP startup settings manually, run the following example in Windows PowerShell:

```
PowerShell

Get-Service -Name Dhcp
Where-Object StartType -ne Automatic
Set-Service -StartupType Automatic
```

4. Disable VMware Tools

Make “VMware Tools” service start-type to disabled if it exists as they aren't required for the VM in Azure.

ⓘ Note

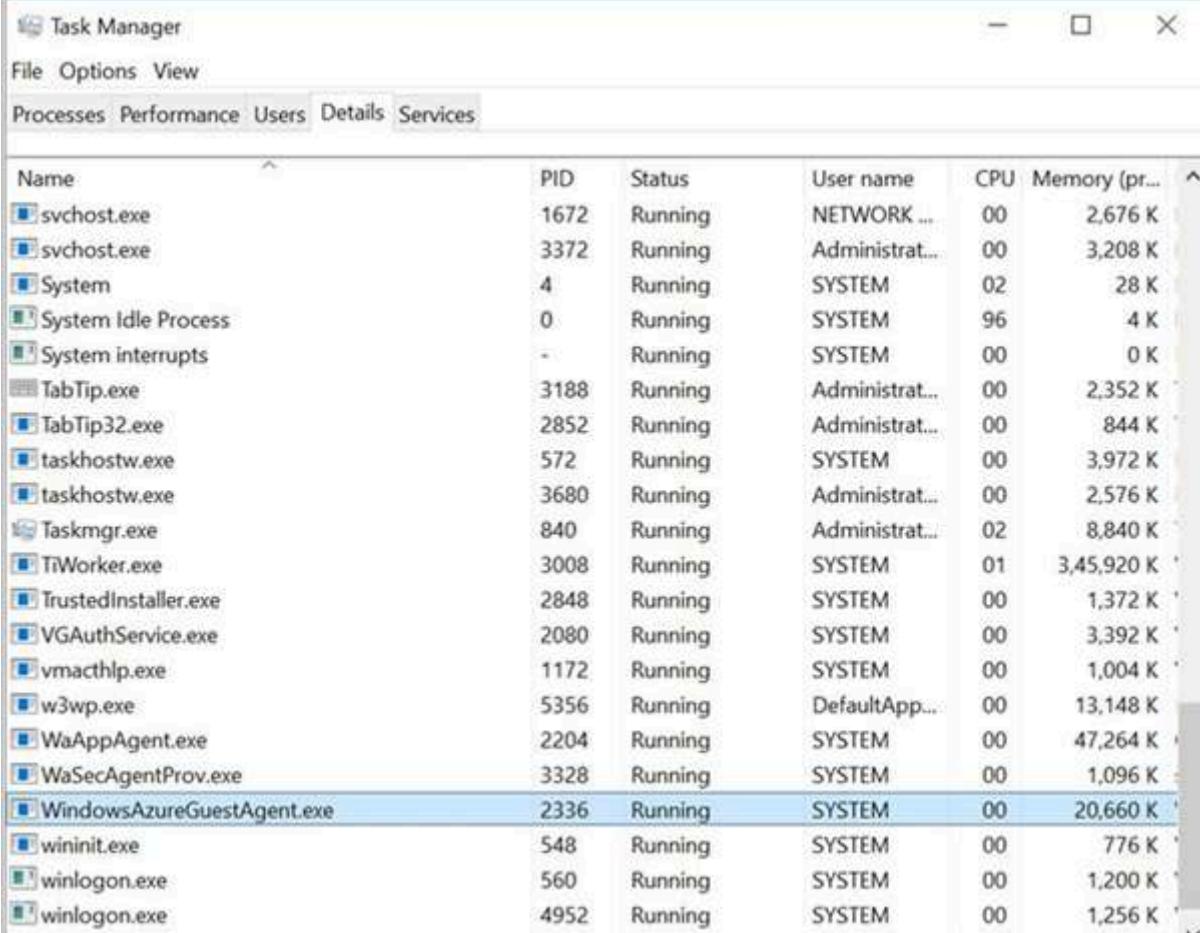
To connect to Windows Server 2003 VMs, Hyper-V Integration Services must be installed on the Azure VM. Windows Server 2003 machines don't have this installed by default. See this [article](#) to install and prepare for migration.

5. Install the Windows Azure Guest Agent

Azure Migrate will attempt to install the Microsoft Azure Virtual Machine Agent (VM Agent), a secure, lightweight process that manages virtual machine (VM) interaction with the Azure Fabric Controller. The VM Agent has a primary role in enabling and executing Azure virtual machine extensions that enable post-deployment configuration of VM, such as installing and configuring software. Azure Migrate automatically installs the Windows VM agent on Windows Server 2008 R2 and higher versions.

The Windows VM agent can be manually installed with a Windows installer package. To manually install the Windows VM Agent, [download the VM Agent installer](#). You can also search for a specific version in the [GitHub Windows IaaS VM Agent releases](#). The VM Agent is supported on Windows Server 2008 (64 bit) and later.

To check if the Azure VM Agent was successfully installed, open Task Manager, select the **Details** tab, and look for the process name *WindowsAzureGuestAgent.exe*. The presence of this process indicates that the VM agent is installed. You can also use [PowerShell to detect the VM agent](#).



Name	PID	Status	User name	CPU	Memory (pr...)
svchost.exe	1672	Running	NETWORK...	00	2,676 K
svchost.exe	3372	Running	Administrat...	00	3,208 K
System	4	Running	SYSTEM	02	28 K
System Idle Process	0	Running	SYSTEM	96	4 K
System interrupts	-	Running	SYSTEM	00	0 K
TabTip.exe	3188	Running	Administrat...	00	2,352 K
TabTip32.exe	2852	Running	Administrat...	00	844 K
taskhostw.exe	572	Running	SYSTEM	00	3,972 K
taskhostw.exe	3680	Running	Administrat...	00	2,576 K
Taskmgr.exe	840	Running	Administrat...	02	8,840 K
TiWorker.exe	3008	Running	SYSTEM	01	3,45,920 K
TrustedInstaller.exe	2848	Running	SYSTEM	00	1,372 K
VGAuthService.exe	2080	Running	SYSTEM	00	3,392 K
vmacthlp.exe	1172	Running	SYSTEM	00	1,004 K
w3wp.exe	5356	Running	DefaultApp...	00	13,148 K
WaAppAgent.exe	2204	Running	SYSTEM	00	47,264 K
WaSecAgentProv.exe	3328	Running	SYSTEM	00	1,096 K
WindowsAzureGuestAgent.exe	2336	Running	SYSTEM	00	20,660 K
wininit.exe	548	Running	SYSTEM	00	776 K
winlogon.exe	560	Running	SYSTEM	00	1,200 K
winlogon.exe	4952	Running	SYSTEM	00	1,256 K

After the aforementioned changes are performed, the system partition will be unloaded. The VM is now ready for migration. [Learn more about the changes for Windows servers](#).

Changes performed on Linux servers

1. Discover and mount Linux OS partitions

Before performing relevant configuration changes, the preparation script will validate if the correct OS disk was selected for migration. The script will collect information on all partitions, their UUIDs, and mount points. The script will look through all these visible partitions to locate the /boot and /root partitions.

The following actions are performed in this step:

- Discover /root partition:
 - Mount each visible partition and look for etc/fstab.

- If the fstab files aren't found, the partition is unmounted, and the search continues for the correct partition.
- If the fstab files are found, read the fstab content to identify the root device and mount it as the base mount point.
- Discover /boot and other system partitions:
 - Use fstab content to determine if /boot is a separate partition. If it's a separate partition, then obtain the boot partition device name from the fstab content or look for the partition, which has the boot flag.
 - The script will proceed to discover and mount /boot, and other necessary partitions on "/mnt/azure_sms_root" to build the root filesystem tree required for chroot jail. Other necessary partitions include: /boot/grub/menu.lst, /boot/grub/grub.conf, /boot/grub2/grub.cfg, /boot/grub/grub.cfg, /boot/efi (for UEFI boot), /var, /lib, /etc, /usr, and others.

2. Discover OS Version

Once the root partition is discovered, the script will use the following files to determine the Linux Operating System distribution and version.

- RHEL: etc/redhat-release
- OL: etc/oracle-release
- SLES: etc/SuSE-release
- Ubuntu: etc/lsb-release
- Debian: etc/debian_version

3. Install Hyper-V Linux Integration Services and regenerate kernel image

The next step is to inspect the kernel image and rebuild the Linux init image so, that it contains the necessary Hyper-V drivers (**hv_vmbus**, **hv_storvsc**, **hv_netvsc**) on the initial ramdisk. Rebuilding the init image ensures that the VM will boot in Azure.

Azure runs on the Hyper-V hypervisor. So, Linux requires certain kernel modules to run in Azure. To prepare your Linux image, you need to rebuild the initrd so that at least the **hv_vmbus** and **hv_storvsc** kernel modules are available on the initial ramdisk. The mechanism for rebuilding the initrd or initramfs image may vary depending on the distribution. Consult your distribution's documentation or support for the proper procedure. Here's one example for rebuilding the initrd by using the `mkinitrd` utility:

- a. Find the list of kernels installed on the system (`/lib/modules`)
- b. For each module, inspect if the Hyper-V drivers are already included.
- c. If any of these drivers are missing, add the required drivers and regenerate the image for the corresponding kernel version.

⚠ Note

This step may not apply to Ubuntu and Debian VMs as the Hyper-V drivers are built-in by default. [Learn more about the changes.](#)

An illustrative example for rebuilding initrd

- Back up the existing initrd image

Bash

```
cd /boot
sudo cp initrd-`uname -r`.img  initrd-`uname -r`.img.bak
```

- Rebuild the initrd with the hv_vmbus and hv_storvsc kernel modules:

Bash

```
sudo mkinitrd --preload=hv_storvsc --preload=hv_vmbus -v -f initrd-
`uname -r`.img `uname -r`
```

Most new versions of Linux distributions have this included by default. If not included, install manually for all versions except those called out, using the aforementioned steps.

4. Enable Azure Serial Console logging

The script will then make changes to enable Azure Serial Console logging. Enabling console logging helps with troubleshooting issues on the Azure VM. Learn more about Azure Serial Console for Linux [Azure Serial Console for Linux - Virtual Machines | Microsoft Docs](#).

Modify the kernel boot line in GRUB or GRUB2 to include the following parameters, so that all console messages are sent to the first serial port. These messages can assist Azure support with debugging any issues.

config

```
console=ttyS0,115200n8 earlyprintk=ttyS0,115200 rootdelay=300
```

We also recommend removing the following parameters if they exist.

config

```
rhgb quiet crashkernel=auto
```

Refer to this article for specific changes.

5. Network changes for connectivity

Based on the OS Version, the script will perform the required network changes for connectivity to the migrated VM. The changes include:

- a. Move (or remove) the udev rules to avoid generating static rules for the Ethernet interface. These rules cause problems when you clone a virtual machine in Azure.

An illustrative example for RedHat servers

Bash

```
sudo ln -s /dev/null /etc/udev/rules.d/75-persistent-net-generator.rules
sudo rm -f /etc/udev/rules.d/70-persistent-net.rules
```

- b. Remove Network Manager if necessary. Network Manager can interfere with the Azure Linux agent for a few OS versions. It's recommended to make these changes for servers running RedHat and Ubuntu distributions.
- c. Uninstall this package by running the following command:

An illustrative example for RedHat servers

Bash

```
sudo rpm -e --nodeps NetworkManager
```

- d. Backup existing NIC settings and create eth0 NIC configuration file with DHCP settings. To do this, the script will create or edit the /etc/sysconfig/network-scripts/ifcfg-eth0 file, and add the following text:

An illustrative example for RedHat servers

config

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
TYPE=Ethernet
USERCTL=no
PEERDNS=yes
IPV6INIT=no
```

```
PERSISTENT_DHCLIENT=yes  
NM_CONTROLLED=yes
```

e. Reset etc/sysconfig/network file as follows:

An illustrative example for RedHat servers

```
config  
  
NETWORKING=yes  
HOSTNAME=localhost.localdomain
```

6. Fstab validation

Azure Migrate will validate the entries of the fstab file and replace fstab entries with persistent volume identifiers, UUIDs wherever needed. This ensures the drive/partition name remains constant no matter the system it's attached to.

- If the device name is a standard device name (say /dev/sdb1), then:
 - If it's a root or boot partition, then it's replaced with UUID.
 - If the partition coexists with either the root or boot partition as standard partitions on the same disk, then it's replaced with UUID.
- If the device name is UUID/LABEL/LV, then no changes will be done.
- If it's a network device (nfs, cifs, smbfs, and etc), then the script will comment the entry. To access it, you can uncomment the same and reboot your Azure VM.

7. Install the Linux Azure Guest Agent

Azure Migrate will attempt to install the Microsoft Azure Linux Agent (waagent), a secure, lightweight process that manages Linux & FreeBSD provisioning, and VM interaction with the Azure Fabric Controller. [Learn more](#) about the functionality enabled for Linux and FreeBSD IaaS deployments via the Linux agent.

Review the list of [required packages](#) to install Linux VM agent. Azure Migrate installs the Linux VM agent automatically for RHEL 9.x, 8.x/7.x/6.x, Ubuntu 14.04/16.04/18.04/19.04/19.10/20.04, SUSE 15 SP0/15 SP1/12, Debian 9/8/7, and Oracle 7/6 when using the agentless method of VMware migration. Follow these instructions to [install the Linux Agent manually](#) for other OS versions.

You can use the command to verify the service status of the Azure Linux Agent to make sure it's running. The service name might be **walinuxagent** or **waagent**. Once the hydration changes are done, the script will unmount all the partitions mounted, deactivate volume groups, and then flush the devices.

Bash

```
sudo vgchange -an <vg-name>
sudo lockdev -flushbufs <disk-device-name>
```

[Learn more on the changes for Linux servers.](#)

Clean up the temporary VM

After the necessary changes are performed, Azure Migrate will spin down the temporary VM and free the attached OS disks (and data disks). This marks the end of the *hydration process*.

After this, the modified OS disk and the data disks that contain the replicated data are cloned. A new virtual machine is created in the target region, virtual network, and subnet, and the cloned disks are attached to the virtual machine. This marks the completion of the migration process.

Learn more

- [Prepare on-premises machines for migration to Azure.](#)

Agent-based migration architecture

05/13/2025

This article provides an overview of the architecture and processes used for agent-based replication of VMware VMs with the [Migration and modernization](#) tool.

Using the Migration and modernization tool, you can replicate VMware VMs with a couple of options:

- Migrate VMs using agent-based replication, as described in this article.
- Migrate VMware VMs using agentless replication. This migrates VMs without needing to install anything on them.

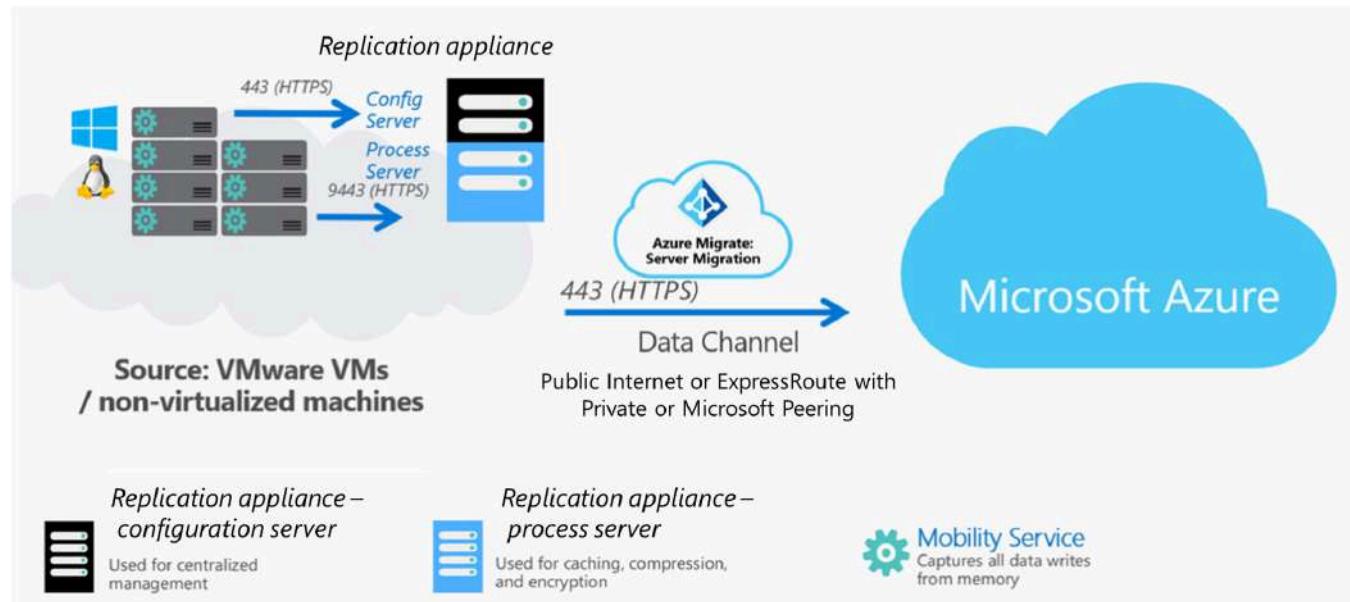
Learn more about [selecting and comparing](#) migration methods for VMware VMs.

Agent-based migration

Agent-based migration is used to migrate on-premises VMware VMs and physical servers to Azure. It can also be used to migrate other on-premises virtualized servers, as well as private and public cloud VMs, including AWS instances, and GCP VMs. Agent-based migration in Azure Migrate uses some backend functionality from the [Azure Site Recovery](#) service.

Architectural components

The diagram illustrates the components involved in agent-based migration.



The table summarizes the components used for agent-based migration.

Component	Details	Installation
Replication appliance	<p>The replication appliance (configuration server/process server) is an on-premises server that acts as a bridge between the on-premises environment, and the Migration and modernization tool. The appliance discovers the on-premises server inventory, so that the Migration and modernization tool can orchestrate replication and migration. The appliance has two components:</p> <p>Configuration server: Connects to the Migration and modernization tool and coordinates replication.</p> <p>Process server: Handles data replication. The process server receives server data, compresses, and encrypts it, and sends to Azure. In Azure, the Migration and modernization tool writes the data to managed disks.</p>	By default the process server is installed together with the configuration server on the replication appliance.
Mobility service	<p>The Mobility service is an agent installed on each server you want to replicate and migrate. It sends replication data from the server to the process server.</p>	Installation files for different versions of the Mobility service are located on the replication appliance. You download and install the agent you need, according to the operating system and version of the server you want to replicate.

Mobility service installation

You can deploy the Mobility Service using the following methods:

- **Push installation:** The process server installs the Mobility service when you enable protection for a server.
- **Install manually:** You can install the Mobility service manually on each server through UI or command prompt.

The Mobility service communicates with the replication appliance and replicated servers. If you have antivirus software running on the replication appliance, process servers, or servers being replicated, the following folders should be excluded from scanning:

- C:\Program Files\Microsoft Azure Recovery Services Agent
- C:\ProgramData\ASR
- C:\ProgramData\ASRLogs

- C:\ProgramData\ASRSetupLogs
- C:\ProgramData\LogUploadServiceLogs
- C:\ProgramData\Microsoft Azure Site Recovery
- C:\Program Files (x86)\Microsoft Azure Site Recovery
- C:\ProgramData\ASR\agent (on Windows servers with the Mobility service installed)

Replication process

1. When you enable replication for a server, initial replication to Azure begins.
2. During initial replication, the Mobility service reads data from the server disks, and sends it to the process server.
3. This data is used to seed a copy of the disk in your Azure subscription.
4. After initial replication finishes, replication of delta changes to Azure begins. Replication is block-level, and near-continuous.
5. The Mobility service intercepts writes to disk memory, by integrating with the storage subsystem of the operating system. This method avoids disk I/O operations on the replicating server, for incremental replication.
6. Tracked changes for a server are sent to the process server on inbound port HTTPS 9443. This port can be modified. The process server compresses and encrypts it, and sends it to Azure.

Ports

[Expand table](#)

Device	Connection
Replicating servers	<p>The Mobility service running on VMs communicates with the on-premises replication appliance on port HTTPS 443 inbound, for replication management.</p> <p>Servers send replication data to the process server on port HTTPS 9443 inbound. This port can be modified.</p>
Replication appliance	The replication appliance orchestrates replication with Azure over port HTTPS 443 outbound.
Process server	The process server receives replication data, optimizes, and encrypts it, and sends it to Azure storage over port 443 outbound.

Performance and scaling

By default, you deploy a single replication appliance that runs both the configuration server and the process server. If you're only replicating a few servers, this deployment is sufficient. However, if you're replicating and migrating hundreds of servers, a single process server might not be able to handle all the replication traffic. In this case, you can deploy additional scale-out process servers.

Plan VMware deployment

If you're replicating VMware VMs, you can use the [Site Recovery Deployment Planner for VMware](#), to help determine performance requirements, including the daily data change rate, and the process servers you need.

Replication appliance capacity

Use the values in this table to figure out whether you need an additional process server in your deployment.

- If the daily change rate (churn rate) is over 2 TB, deploy an additional process server.
- If you're replicating more than 200 servers, deploy an additional replication appliance.

[[Expand table

CPU	Memory	Free space-data caching	Churn rate	Replication limits
8 vCPUs (2 sockets * 4 cores @ 2.5 GHz)	16 GB	300 GB	500 GB or less	< 100 servers
12 vCPUs (2 sockets * 6 cores @ 2.5 GHz)	18 GB	600 GB	501 GB to 1 TB	100-150 servers.
16 vCPUs (2 sockets * 8 cores @ 2.5 GHz)	32 GB	1 TB	1 TB to 2 TB	151-200 servers.

Sizing scale-out process servers

If you need to deploy a scale-out process server, use this table to figure out server sizing.

[[Expand table

Process server	Free space for data caching	Churn rate	Replication limits
4 vCPUs (2 sockets * 2 cores @ 2.5 GHz), 8-GB memory	300 GB	250 GB or less	Up to 85 servers
8 vCPUs (2 sockets * 4 cores @ 2.5 GHz), 12-GB memory	600 GB	251 GB to 1 TB	86-150 servers.
12 vCPUs (2 sockets * 6 cores @ 2.5 GHz), 24-GB memory	1 TB	1-2 TB	151-225 servers.

Throttle upload bandwidth.

VMware traffic that replicates to Azure goes through a specific process server. You can limit upload throughput by throttling bandwidth on the servers that are running as process servers. You can influence bandwidth using this registry key:

- The HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Azure Backup\Replication\UploadThreadsPerVM registry value specifies the number of threads that are used for data transfer (initial or delta replication) of a disk. A higher value increases the network bandwidth that's used for replication. The default value is four. The maximum value is 32. To optimize the value, monitor the traffic.
- In addition, you can throttle bandwidth on the process server as follows:
 1. On the process server, open the Azure Backup MMC snap-in. There's a shortcut on the desktop or in the folder C:\Program Files\Microsoft Azure Recovery Services Agent\bin.
 2. In the snap-in, select **Change Properties**.
 3. In **Throttling**, select **Enable internet bandwidth usage throttling for backup operations**. Set the limits for work and non-work hours. Valid ranges are from 512 Kbps to 1,023 Mbps.

Next steps

Try out agent-based migration for [VMware](#)

Networking architecture design

Article • 12/12/2024

This article provides information about sample architectures, solutions, and guides that can help you explore networking in Azure.

Designing and implementing Azure networking capabilities is a critical part of your cloud solution. You'll need to make networking design decisions to properly support your workloads and services.

Azure provides a wide range of networking tools and capabilities. These are just some of the key networking services available in Azure:

- [Azure Virtual Network](#). Provision private networks, and optionally connect to on-premises datacenters.
- [Azure Virtual WAN](#). Optimize and automate branch-to-branch connectivity.
- [Azure Private Link](#). Enable private access to services that are hosted on the Azure platform while keeping your data on the Microsoft network.
- [Azure Firewall](#). Provide protection for your Azure Virtual Network resources.
- [Azure Application Gateway](#). Build highly secure, scalable, highly available web front ends.
- [Azure ExpressRoute](#). Create a fast, reliable, and private connection to Azure.
- [Azure Load Balancer](#). Deliver high availability and network performance to your apps.
- [Azure VPN Gateway](#). Establish high security cross-premises connectivity.

For information about more Azure networking services, see [Azure networking](#).

Introduction to networking on Azure

If you're new to networking on Azure, the best way to learn more is with [Microsoft Learn training](#), a free online training platform. Microsoft Learn provides interactive training for Microsoft products and more.

Here's a good introduction to Azure networking:

- [Explore Azure networking services](#)

And here's a comprehensive learning path:

- [Configure and manage virtual networks for Azure administrators](#)

Path to production

Consider these technologies and solutions as you plan and implement your deployment:

- [Azure Private Link in a hub-and-spoke network](#)
- [Recommendations for using availability zones and regions](#)
- [Choose between virtual network peering and VPN gateways](#)
- [Use Azure ExpressRoute with Microsoft Power Platform](#)

Best practices

The [Azure Well-Architected Framework](#) is a set of guiding tenets, based on five pillars, that you can use to improve the quality of your architectures. These articles apply the pillars to the use of some Azure networking services:

- [Review of Azure Application Gateway](#)
- [Review of Azure Firewall](#)

The [Cloud Adoption Framework](#) is a collection of documentation, implementation guidance, best practices, and tools that are designed to accelerate your cloud adoption. You might find these articles helpful as you plan and implement your networking solution:

- [Connectivity to other cloud providers](#)
- [Connectivity to Oracle Cloud Infrastructure](#)

Networking architectures

The following sections, organized by category, provide links to sample networking architectures.

High availability

- [Deploy highly available NVAs](#)
- [Multi-tier web application built for HA/DR](#)

Hybrid networking

- [Design a hybrid Domain Name System solution with Azure](#)
- [Hybrid availability and performance monitoring](#)
- [Implement a secure hybrid network](#)

Hub-and-spoke topology

- Hub-and-spoke network topology in Azure
- Hub-and-spoke network topology with Azure Virtual WAN

Virtual WAN

- Global transit network architecture and Virtual WAN
- Interconnect with China using Azure Virtual WAN and Secure Hub
- Migrate to Azure Virtual WAN
- SD-WAN connectivity architecture with Azure Virtual WAN
- Virtual WAN network topology (Microsoft-managed)
- Virtual WAN architecture optimized for department-specific requirements
- Hub-and-spoke network topology with Azure Virtual WAN

Multi-region networking

- Multi-region load balancing with Azure Traffic Manager and Application Gateway

Stay current with networking

Get the [latest updates on Azure networking products and features](#).

Additional resources

Example solutions

These are some additional sample networking architectures:

- Traditional Azure networking topology
- What is an Azure landing zone?
- Multitenant SaaS on Azure
- Baseline highly available zone-redundant web application
- Network topology and connectivity for Azure VMware Solution
- Private Link and DNS integration at scale
- Trusted Internet Connection (TIC) 3.0 compliance for internet-facing applications
- Update route tables by using Azure Route Server

AWS or Google Cloud professionals

These articles provide service mapping and comparison between Azure and other cloud services. They can help you ramp up quickly on Azure.

- [Compare AWS and Azure networking options](#)
 - [Google Cloud to Azure services comparison - Networking](#)
-

Feedback

Was this page helpful?

 Yes

 No

Security architecture design

Microsoft Entra ID

Azure Firewall

Azure Front Door

Azure Key Vault

Azure Private Link

Information security has always been a complex subject, and it evolves quickly with the creative ideas and implementations of attackers and security researchers.

Security is one of the most important aspects of any architecture. Good security provides confidentiality, integrity, and availability assurances against deliberate attacks and abuse of your valuable data and systems. Losing these assurances can harm your business operations and revenue, and your organization's reputation.

! Note

Learn how cloud security is an ongoing journey of incremental progress and maturity, in [Security in the Microsoft Cloud Adoption Framework for Azure](#). Learn how to build security into your solution, in the Azure Well-Architected Framework [Overview of the security pillar](#).

Here are some broad categories to consider when you design a security system:



Azure provides a wide range of security tools and capabilities. These are just some of the key security services available in Azure:

- [Microsoft Defender for Cloud](#) . A unified infrastructure security management system that strengthens the security posture of your datacenters. It also provides advanced

threat protection across your hybrid workloads in the cloud and on-premises.

- [Microsoft Entra ID](#). The Microsoft cloud-based identity and access management service.
- [Azure Front Door](#). A global, scalable entry-point that uses the Microsoft global edge network to create fast, highly secure, and widely scalable web applications.
- [Azure Firewall](#). A cloud-native, intelligent network firewall security service that provides threat protection for your cloud workloads that run in Azure.
- [Azure Key Vault](#). A high-security secret store for tokens, passwords, certificates, API keys, and other secrets. You can also use Key Vault to create and control the encryption keys used to encrypt your data.
- [Azure Private Link](#). A service that enables you to access Azure PaaS services, Azure-hosted services that you own, or partner services over a private endpoint in your virtual network.
- [Azure Application Gateway](#). An advanced web traffic load balancer that enables you to manage traffic to your web applications.
- [Azure Policy](#). A service that helps you enforce organizational standards and assess compliance.

For a more comprehensive description of Azure security tools and capabilities, see [End-to-end security in Azure](#).

Introduction to security on Azure

If you're new to security on Azure, the best way to learn more is with [Microsoft Learn training](#). This free online platform provides interactive training for Microsoft products and more.

Here are two learning paths to get you started:

- [Microsoft Azure Fundamentals: Describe general security and network security features](#)
- [Microsoft Security, Compliance, and Identity Fundamentals: Describe the capabilities of Microsoft security solutions](#)

Path to production

- To secure Azure application workloads, you use protective measures like authentication and encryption in the applications themselves. You can also add security layers to the virtual machine (VM) networks that host the applications. See [Firewall and Application Gateway for virtual networks](#) for an overview.
- Zero Trust is a proactive, integrated approach to security across all layers of the digital estate. It explicitly and continuously verifies every transaction, asserts least privilege, and relies on intelligence, advanced detection, and real-time response to threats.

- For an implementation strategy for web apps, see [Zero Trust network for web applications with Azure Firewall and Application Gateway](#).
- Azure governance establishes the tooling needed to support cloud governance, compliance auditing, and automated guardrails. See [Azure governance design area guidance](#) for information about governing your Azure environment.

Best practices

The Azure Well-Architected Framework is a set of guiding tenets, based on five pillars, that you can use to improve the quality of your architectures. For more information, see [Overview of the security pillar](#) and [Security design principles in Azure](#).

The Well-Architected Framework also provides these checklists:

- Azure identity and access management considerations
- Network security
- Data protection considerations
- Governance, risk, and compliance

Security architectures

Identity and access management

- [Microsoft Entra identity management and access management for AWS](#)

Threat protection

- Multilayered protection for Azure virtual machine access

Stay current with security

Get the latest updates on [Azure security services and features](#) .

Additional resources

Example solutions

- Improved-security access to multitenant web apps from an on-premises network
- Restrict interservice communications

- Securely managed web applications
- Web app private connectivity to Azure SQL database

[Browse all our security architectures.](#)

AWS or Google Cloud professionals

- [Identity with Azure and AWS](#)
- [AWS to Azure services comparison - Security](#)
- [Google Cloud to Azure services comparison - Security](#)

Next steps

Security architecture is part of a comprehensive set of security guidance that also includes:

- [Security in the Microsoft Cloud Adoption Framework for Azure](#): A high-level overview of a cloud security end state.
- [Azure Well-Architected Framework](#): Guidance on securing your workloads on Azure.
- [Azure security benchmarks](#): Prescriptive best practices and controls for Azure security.
- [End-to-end security in Azure](#): Documentation that introduces you to the security services in Azure.
- [Top 10 security best practices for Azure](#): Top Azure security best practices that Microsoft recommends based on lessons learned across customers and our own environments.
- [Microsoft Cybersecurity Architectures](#): The diagrams describe how Microsoft security capabilities integrate with Microsoft platforms and 3rd-party platforms.

Azure Firewall and Application Gateway for virtual networks

Azure Application Gateway

Azure Firewall

Azure Front Door

Azure Virtual Network

Azure Web Application Firewall

To help secure Azure application workloads, use protective measures such as authentication and encryption in the applications themselves. You can add security layers to the virtual networks that host the applications. These security layers help protect the application's inbound flows from unintended use. They also limit outbound flows to the internet to only those endpoints that your application requires. This article describes [Azure Virtual Network](#) security services like Azure DDoS Protection, Azure Firewall, and Azure Application Gateway. It also describes when to use each service and network design options that combine them.

- [DDoS Protection](#), combined with application design best practices, provides enhanced DDoS mitigation features that improve defense against DDoS attacks. You should enable DDoS Protection on every perimeter virtual network.
- [Azure Firewall](#) is a managed, next-generation firewall that provides [network address translation \(NAT\)](#) capabilities. Azure Firewall filters packets based on IP addresses and Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) ports. It can filter traffic by using application-based attributes, such as HTTP(S) and SQL. Azure Firewall also applies Microsoft threat intelligence to help identify malicious IP addresses. For more information, see [Azure Firewall documentation](#).
- [Azure Firewall Premium](#) includes all the functionality of Azure Firewall Standard, in addition to features like transport layer security (TLS) inspection and intrusion detection and prevention system (IDPS).
- [Application Gateway](#) is a managed web traffic load balancer and HTTP(S) full reverse proxy that can perform Secure Socket Layer (SSL) encryption and decryption. Application Gateway preserves the original client IP address in an `X-Forwarded-For` HTTP header. Application Gateway also uses Azure Web Application Firewall to inspect web traffic and detect attacks at the HTTP layer. For more information, see [Application Gateway documentation](#).
- [Web Application Firewall](#) is an optional addition to Application Gateway. It inspects HTTP requests and prevents web layer attacks, such as SQL injection and cross-site scripting. For more information, see [Web Application Firewall documentation](#).

These Azure services complement each other. Depending on your needs, using one service might suit your workloads better. However, you can use these services together to help provide

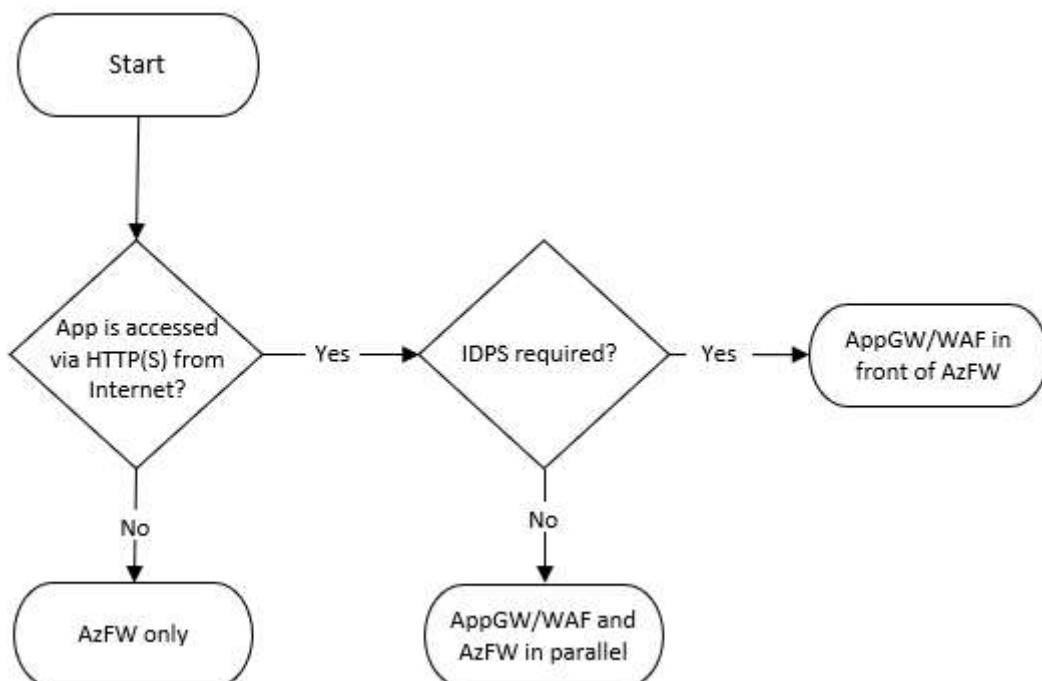
optimal protection at both the network and application layers. Use the following decision tree and the examples in this article to choose the best security option for your application's virtual network.

Azure Firewall and Application Gateway use different technologies to help secure different types of data flows.

[Expand table](#)

Application flow	Can be filtered by Azure Firewall	Can be filtered by Web Application Firewall on Application Gateway
HTTP(S) traffic from on-premises or internet to Azure (inbound)	Yes	Yes
HTTP(S) traffic from Azure to on-premises or internet (outbound)	Yes	No
Non-HTTP(S) traffic (inbound or outbound)	Yes	No

The design can vary for each application based on the network flows that it requires. The following diagram provides a simplified decision tree that helps you choose the recommended approach for your application. This choice depends on whether the application is published via HTTP(S) or some other protocol.



This article describes the widely recommended designs shown in the flow chart and designs suited for less common scenarios:

- **Azure Firewall only:** Use this design when there are no web applications in the virtual network. It controls both inbound traffic to the applications and outbound traffic.
- **Application Gateway only:** Use this design when only web applications are in the virtual network and [network security groups \(NSGs\)](#) provide sufficient output filtering. Azure Firewall provides functionality that can help prevent several attack scenarios, such as data exfiltration and IDPS. As a result, the Application Gateway-only design isn't usually recommended, so it isn't included in the previous flow chart.
- **Azure Firewall and Application Gateway in parallel:** Use this design when you want Application Gateway to protect HTTP(S) applications from web attacks and Azure Firewall to protect all other workloads and filter outbound traffic. Azure Firewall and Application Gateway in parallel is a common design.
- **Application Gateway in front of Azure Firewall:** Use this design when you want Azure Firewall to inspect all traffic, Web Application Firewall to protect web traffic, and the application to identify the client's source IP address. With Azure Firewall Premium and TLS inspection, this design also supports the end-to-end SSL scenario.
- **Azure Firewall in front of Application Gateway:** Use this design when you want Azure Firewall to inspect and filter traffic before it reaches the Application Gateway. Because Azure Firewall doesn't decrypt HTTPS traffic, its added functionality to the Application Gateway is limited. This scenario isn't documented in the previous flow chart.

Variations of these fundamental designs are described later in this article and include:

- [On-premises application clients](#).
- [Hub-and-spoke networks](#).
- [Azure Kubernetes Service \(AKS\) implementations](#).

You can add other reverse proxy services, like an [Azure API Management](#) gateway or [Azure Front Door](#). Or you can replace the Azure resources with non-Microsoft [network virtual appliances \(NVAs\)](#).

Note

In the following scenarios, an Azure virtual machine (VM) is used as an example of a web application workload. These scenarios are also valid for other workload types, such as containers or Azure Web Apps. For setups that include private endpoints, consider the

recommendations in [Azure Firewall scenarios to inspect traffic destined to a private endpoint](#).

Azure Firewall-only design

If there are no web-based workloads in the virtual network that can benefit from Web Application Firewall, you can use the Azure Firewall-only design. The design in this example is simple, but you can review the packet flow to better understand more complex designs. In this design, all inbound traffic is sent to Azure Firewall via user-defined routes (UDRs) for connections from on-premises or other Azure virtual networks. It's addressed to the Azure Firewall public IP address for connections from the public internet, as shown in the following diagram. UDRs direct outbound traffic from Azure virtual networks to Azure Firewall, as shown in the following dialog.

The following table summarizes the traffic flows for this scenario.

 [Expand table](#)

Flow	Goes through Application Gateway/Web Application Firewall	Goes through Azure Firewall
HTTP(S) traffic from the internet or on-premises to Azure	N/A	Yes
HTTP(S) traffic from Azure to the internet or on-premises	N/A	Yes
Non-HTTP(S) traffic from the internet or on-premises to Azure	N/A	Yes
Non-HTTP(S) traffic from Azure to the internet or on-premises	N/A	Yes

Azure Firewall doesn't inspect inbound HTTP(S) traffic. But it can apply layer 3 and layer 4 rules and fully qualified domain name (FQDN)-based application rules. Azure Firewall inspects outbound HTTP(S) traffic, depending on the Azure Firewall tier and whether you configure TLS inspection:

- Azure Firewall Standard inspects only layer 3 and layer 4 attributes of packets in network rules and the Host HTTP header in application rules.

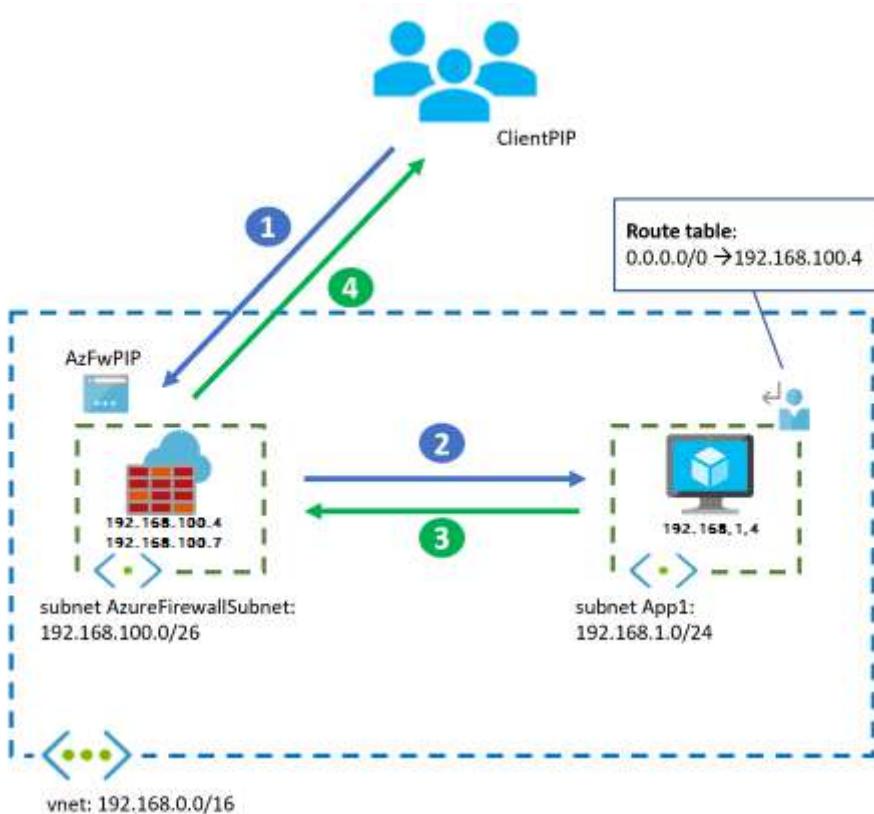
- Azure Firewall Premium adds capabilities, such as inspecting other HTTP headers (like the user-agent) and enabling TLS inspection for deeper packet analysis. However, Azure Firewall isn't the same as Web Application Firewall. If you have web workloads in your virtual network, we recommend that you use Web Application Firewall.

The following packet walk example shows how a client accesses a VM-hosted application from the public internet. The diagram includes only one VM for simplicity. For higher availability and scalability, there are multiple application instances behind a load balancer. In this design, Azure Firewall inspects incoming connections from the public internet and outbound connections from the application subnet VM by using the UDR.

- In this example, Azure Firewall automatically deploys several instances with the front-end IP address `192.168.100.4` and internal addresses within the range `192.168.100.0/26`. Normally, these instances aren't visible to the Azure administrator. However, being aware of them can be helpful for troubleshooting network problems.
- If traffic comes from an on-premises virtual private network (VPN) or [Azure ExpressRoute](#) gateway instead of the internet, the client starts the connection to the VM's IP address. It doesn't start the connection to the firewall's IP address, and the firewall doesn't do source NAT by default.

Architecture

The following diagram shows the traffic flow and assumes that the instance IP address is `192.168.100.7`.



Workflow

1. The client starts the connection to the public IP address of Azure Firewall.
 - Source IP address: `ClientPIP`
 - Destination IP address: `AzFwPIP`
2. The request to the Azure Firewall public IP address is distributed to a back-end instance of the firewall, which is `192.168.100.7` in this example. The Azure Firewall [Destination Network Address Translation \(DNAT\)](#) rule translates the destination IP address to the application IP address inside the virtual network. Azure Firewall also implements *source network address translation (SNAT)* on the packet if it uses DNAT. For more information, see [Azure Firewall known issues](#). The VM sees the following IP addresses in the incoming packet:
 - Source IP address: `192.168.100.7`
 - Destination IP address: `192.168.1.4`
3. The VM answers the application request, which reverses both the source and destination IP addresses. The inbound flow doesn't require a UDR because the source IP is the Azure Firewall IP address. The UDR in the diagram for `0.0.0.0/0` is for outbound connections to ensure that packets to the public internet go through Azure Firewall.
 - Source IP address: `192.168.1.4`
 - Destination IP address: `192.168.100.7`
4. Azure Firewall undoes the SNAT and DNAT operations and delivers the response to the client.
 - Source IP address: `AzFwPIP`
 - Destination IP address: `ClientPIP`

Application Gateway-only design

This design describes the scenario where only web applications exist in the virtual network, and inspecting outbound traffic with NSGs is sufficient to protect outbound flows to the internet.

Note

We don't recommend this design because using Azure Firewall to control outbound flows, instead of relying solely on NSGs, helps prevent attack scenarios such as data exfiltration. With Azure Firewall, you can help ensure that your workloads only send data to an

approved list of URLs. Also, NSGs operate only at layer 3 and layer 4 and don't support FQDNs.

The key difference from the previous Azure Firewall-only design is that Application Gateway doesn't serve as a routing device with NAT. Instead, it functions as a full reverse application proxy. This approach means that Application Gateway stops the web session from the client and establishes a separate session with one of its back-end servers. Inbound HTTP(S) connections from the internet are sent to the public IP address of Application Gateway, and connections from Azure or on-premises use the gateway's private IP address. Return traffic from the Azure VMs follows standard virtual network routing back to Application Gateway. For more information, see the packet walk later in this article. Outbound internet flows from Azure VMs go directly to the internet.

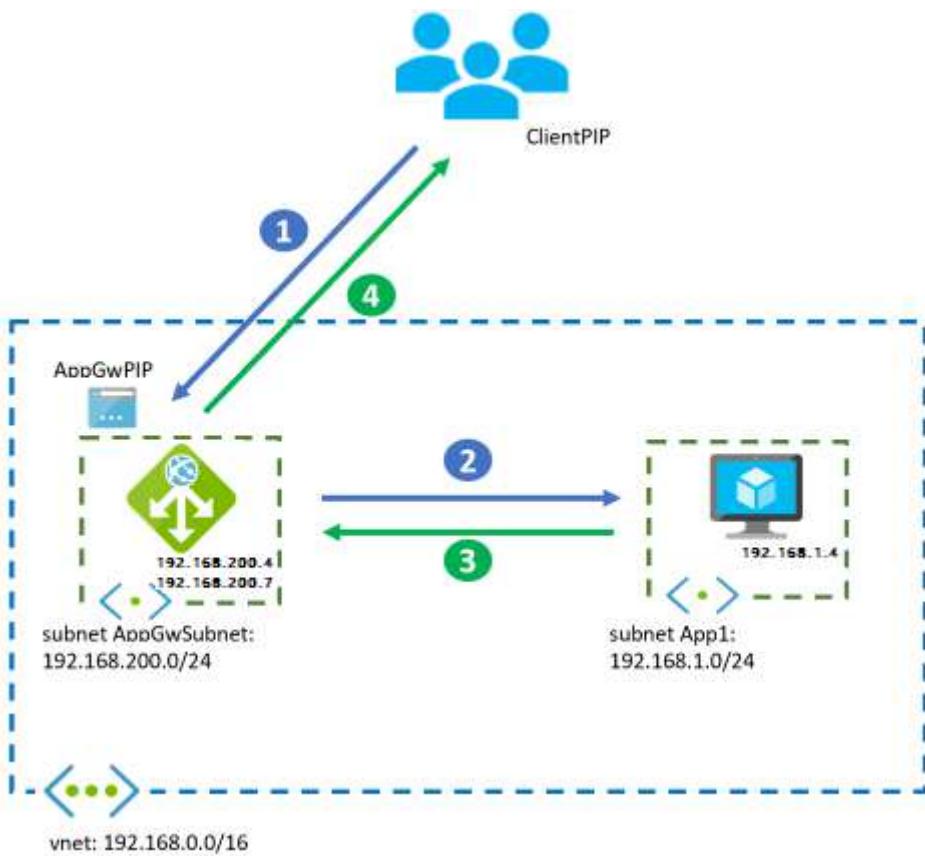
The following table summarizes traffic flows.

[Expand table](#)

Flow	Goes through Application Gateway/Web Application Firewall	Goes through Azure Firewall
HTTP(S) traffic from the internet or on-premises to Azure	Yes	N/A
HTTP(S) traffic from Azure to the internet or on-premises	No	N/A
Non-HTTP(S) traffic from the internet or on-premises to Azure	No	N/A
Non-HTTP(S) traffic from Azure to the internet or on-premises	No	N/A

Architecture

The following packet walk example shows how a client accesses the VM-hosted application from the public internet.



Workflow

1. The client starts the connection to the public IP address of Application Gateway.
 - Source IP address: ClientPIP
 - Destination IP address: AppGwPIP
2. The request to the Application Gateway public IP address is distributed to a back-end instance of the gateway, which is 192.168.200.7 in this example. The Application Gateway instance that receives the request stops the connection from the client and establishes a new connection with one of the back ends. The back end sees the Application Gateway instance as the source IP address. Application Gateway inserts an X-Forwarded-For HTTP header with the original client's IP address.
 - Source IP address, which is the private IP address of the Application Gateway instance: 192.168.200.7
 - Destination IP address: 192.168.1.4
 - X-Forwarded-For header: ClientPIP
3. The VM answers the application request and reverses both the source and destination IP addresses. The VM can reach Application Gateway, so it doesn't need a UDR.
 - Source IP address: 192.168.1.4
 - Destination IP address: 192.168.200.7

4. The Application Gateway instance answers the client.

- Source IP address: AppGwPIP
- Destination IP address: ClientPIP

Application Gateway adds metadata to the packet HTTP headers, such as the `X-Forwarded-For` header that contains the original client's IP address. Some application servers need the source client IP address to serve geolocation-specific content, or for logging. For more information, see [How an application gateway works](#).

- In this example, the IP address `192.168.200.7` is one of the instances deployed by the Application Gateway service automatically. It has the internal, private front-end IP address `192.168.200.4`. These individual instances are normally invisible to the Azure administrator. But noticing the difference can be useful, such as when you troubleshoot network problems.
- The flow is similar if the client comes from an on-premises network over a VPN or ExpressRoute gateway. The difference is the client accesses the private IP address of Application Gateway instead of the public IP address.

 **Note**

For more information about the `X-Forwarded-For` header and how to preserve the host name on a request, see [Preserve the original HTTP host](#).

Azure Firewall and Application Gateway in parallel design

Because of its simplicity and flexibility, it's often best to run Application Gateway and Azure Firewall in parallel.

Implement this design if there's both web and non-web workloads in the virtual network. Web Application Firewall in Application Gateway helps protect inbound traffic to the web workloads. Azure Firewall inspects inbound traffic for the other applications. Azure Firewall covers outbound flows from both workload types.

Inbound HTTP(S) connections from the internet should be sent to the public IP address of Application Gateway. HTTP(S) connections from Azure or on-premises should be sent to its private IP address. Standard virtual network routing sends the packets from Application Gateway to the destination VMs, and from the destination VMs back to Application Gateway. For more information, see the packet walk later in this article.

For inbound non-HTTP(S) connections, traffic should target the public IP address of Azure Firewall if it comes from the public internet. Traffic should be sent through Azure Firewall by UDRs if it comes from other Azure virtual networks or on-premises networks. All outbound flows from Azure VMs are forwarded to Azure Firewall by UDRs.

The following table summarizes the traffic flows for this scenario.

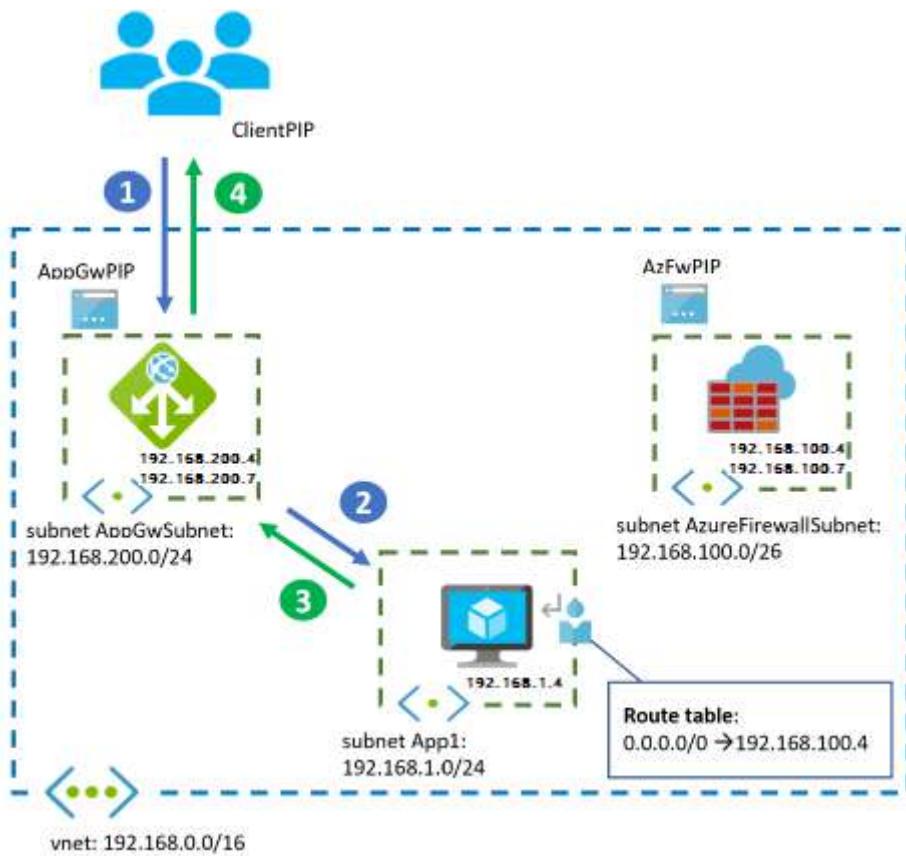
[Expand table](#)

Flow	Goes through Application Gateway/Web Application Firewall	Goes through Azure Firewall
HTTP(S) traffic from the internet or on-premises to Azure	Yes	No
HTTP(S) traffic from Azure to the internet or on-premises	No	Yes
Non-HTTP(S) traffic from the internet or on-premises to Azure	No	Yes
Non-HTTP(S) traffic from Azure to the internet or on-premises	No	Yes

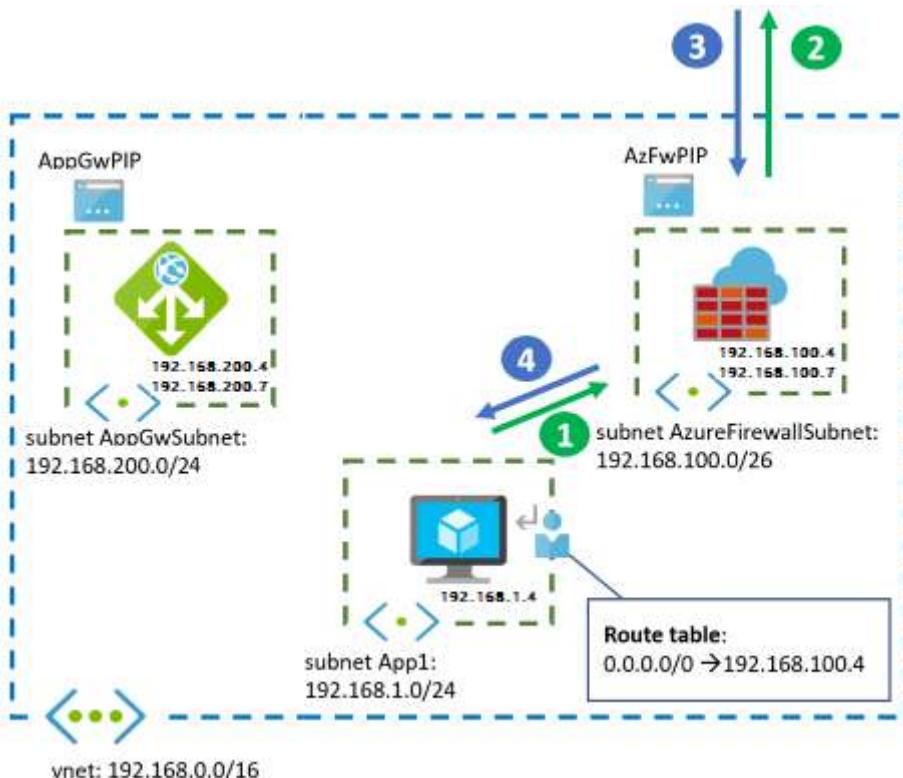
This design provides much more granular egress filtering than solely using NSGs. For example, if applications need connectivity to a specific Azure Storage account, you can use FQDN-based filters. With FQDN-based filters, applications don't send data to rogue storage accounts. If you only use NSGs, you can't prevent this scenario. This design is often used when outbound traffic requires FQDN-based filtering. One scenario is when you [limit egress traffic from an AKS cluster](#).

Architectures

The following diagram illustrates the traffic flow for inbound HTTP(S) connections from an outside client.



The following diagram illustrates the traffic flow for outbound connections from the network VMs to the internet. One example is to connect to back-end systems or get operating system updates.



The packet flow steps for each service are the same as in the previous standalone design options.

Application Gateway in front of Azure Firewall design

This design is explained in more detail in [Zero-trust network for web applications with Azure Firewall and Application Gateway](#). This article focuses on the comparison with the other design options. In this topology, inbound web traffic goes through both Azure Firewall and Web Application Firewall. Web Application Firewall provides protection at the web application layer. Azure Firewall serves as a central logging and control point, and it inspects traffic between Application Gateway and the back-end servers. In this design, Application Gateway and Azure Firewall don't sit in parallel but sit one in front of the other.

With [Azure Firewall Premium](#), this design can support end-to-end scenarios, where Azure Firewall applies TLS inspection to perform IDPS on the encrypted traffic between Application Gateway and the web back end.

This design is suited for applications that need to identify incoming client source IP addresses. For example, it can be used to serve geolocation-specific content or for logging. The Application Gateway in front of Azure Firewall design captures the incoming packet's source IP address in the `X-Forwarded-For` header, so the web server can see the original IP address in this header. For more information, see [How an application gateway works](#).

Inbound HTTP(S) connections from the internet need to be sent to the public IP address of Application Gateway. HTTP(S) connections from Azure or on-premises should be sent to its private IP address. From Application Gateway, UDRs ensure that the packets are routed through Azure Firewall. For more information, see the packet walk later in this article.

For inbound non-HTTP(S) connections, traffic should target the public IP address of Azure Firewall if it comes from the public internet. Traffic should be sent through Azure Firewall by UDRs if it comes from other Azure virtual networks or on-premises networks. All outbound flows from Azure VMs are forwarded to Azure Firewall by UDRs.

An important aspect of this design is that Azure Firewall Premium sees traffic with a source IP address from the Application Gateway subnet. If this subnet is configured with a private IP address (in `10.0.0.0/8`, `192.168.0.0/16`, `172.16.0.0/12`, or `100.64.0.0/10`), Azure Firewall Premium treats traffic from Application Gateway as internal and doesn't apply IDPS rules for inbound traffic. As a result, we recommend that you modify the IDPS private prefixes in the Azure Firewall policy. This modification ensures that the Application Gateway subnet isn't considered an internal source, which allows inbound and outbound IDPS signatures to be applied to the traffic. You can find more information about Azure Firewall IDPS rule directions and private IP prefixes for IDPS in [Azure Firewall IDPS rules](#).

The following table summarizes the traffic flows for this scenario.

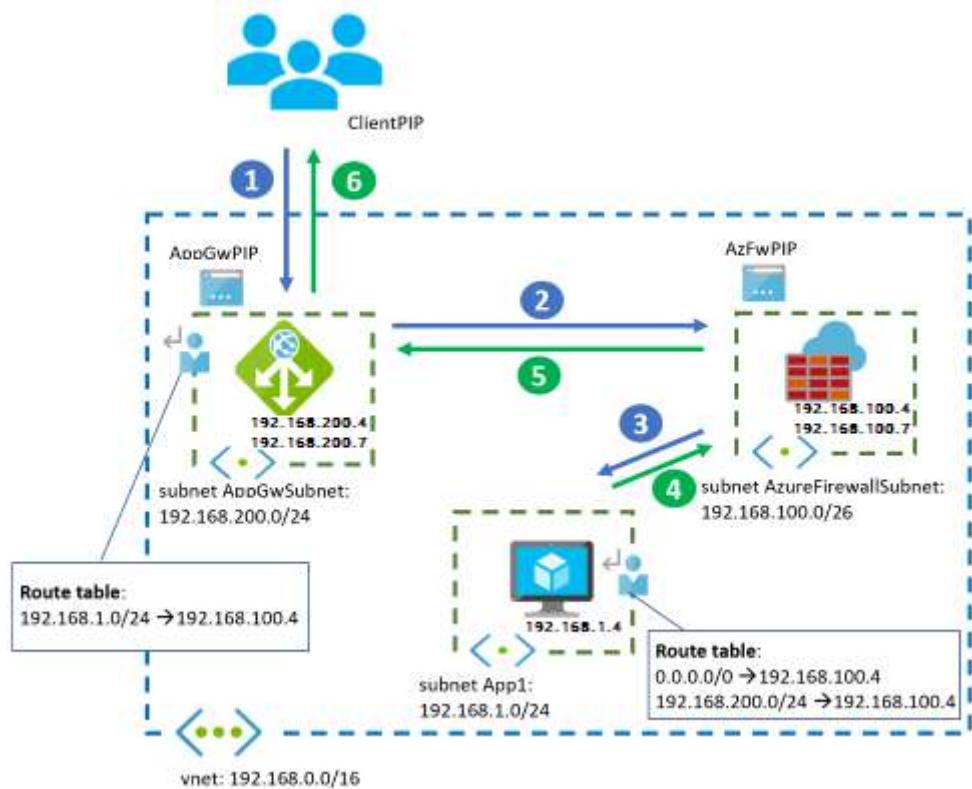
Flow	Goes through Application Gateway/Web Application Firewall	Goes through Azure Firewall
HTTP(S) traffic from the internet or on-premises to Azure	Yes	Yes
HTTP(S) traffic from Azure to the internet or on-premises	No	Yes
Non-HTTP(S) traffic from the internet or on-premises to Azure	No	Yes
Non-HTTP(S) traffic from Azure to the internet or on-premises	No	Yes

For web traffic from on-premises or from the internet to Azure, Azure Firewall inspects flows that Web Application Firewall allows. Depending on whether Application Gateway encrypts back-end traffic, which is traffic from Application Gateway to the application servers, different scenarios can occur:

- Application Gateway encrypts traffic by following zero-trust principles like [end-to-end TLS encryption](#), and Azure Firewall receives encrypted traffic. Azure Firewall Standard can still apply inspection rules, such as layer 3 and layer 4 filtering in network rules, or FQDN filtering in application rules, by using the TLS Server Name Indication (SNI) header. [Azure Firewall Premium](#) provides deeper visibility with TLS inspection, such as URL-based filtering.
- If Application Gateway sends unencrypted traffic to the application servers, Azure Firewall sees inbound traffic in clear text. TLS inspection isn't needed in Azure Firewall.
- If IDPS is enabled in Azure Firewall, it verifies that the HTTP Host header matches the destination IP address. To perform this verification, it needs the name resolution for the FQDN that's specified in the Host header. This name resolution can be performed by using Azure DNS private zones and the default [Azure Firewall DNS settings](#). It can also be achieved with custom DNS servers that need to be configured in the Azure Firewall settings. If you don't have administrative access to the virtual network where Azure Firewall is deployed, the latter method is your only option. One example is with Azure Firewall instances deployed in Azure Virtual WAN-secured hubs.

Architecture

For the rest of the flows, which include inbound non-HTTP(S) traffic and any outbound traffic, Azure Firewall provides IDPS inspection and TLS inspection where suitable. It also provides [FQDN-based filtering in network rules](#) based on DNS.



Workflow

Network traffic from the public internet follows this flow:

1. The client starts the connection to the public IP address of Application Gateway.
 - Source IP address: `ClientPIP`
 - Destination IP address: `AppGwPIP`
2. The request to the Application Gateway public IP address is distributed to a back-end instance of the gateway, which is `192.168.200.7` in this example. The Application Gateway instance stops the connection from the client and establishes a new connection with one of the back ends. The UDR to `192.168.1.0/24` in the Application Gateway subnet forwards the packet to Azure Firewall and preserves the destination IP address to the web application.
 - Source IP address, which is the private IP address of the Application Gateway instance: `192.168.200.7`
 - Destination IP address: `192.168.1.4`

- `X-Forwarded-For` header: `ClientPIP`

3. Azure Firewall doesn't apply SNAT to the traffic because the traffic goes to a private IP address. It forwards the traffic to the application VM if rules allow it. For more information, see [Azure Firewall SNAT private IP address ranges](#). However, if the traffic hits an application rule in the firewall, the workload sees the source IP address of the specific firewall instance that processed the packet because Azure Firewall proxies the connection.

- Source IP address if the traffic is allowed by an Azure Firewall network rule and is the private IP address of one of the Application Gateway instances: `192.168.200.7`
- Source IP address if the traffic is allowed by an Azure Firewall application rule and is the private IP address of one of the Azure Firewall instances: `192.168.100.7`
- Destination IP address: `192.168.1.4`
- `X-Forwarded-For` header: `ClientPIP`

4. The VM answers the request, which reverses both the source and destination IP addresses. The UDR to `192.168.200.0/24` captures the packet sent back to Application Gateway, redirects it to Azure Firewall, and preserves the destination IP address toward Application Gateway.

- Source IP address: `192.168.1.4`
- Destination IP address: `192.168.200.7`

5. Again, Azure Firewall doesn't apply SNAT to the traffic because it goes to a private IP address and forwards the traffic to Application Gateway.

- Source IP address: `192.168.1.4`
- Destination IP address: `192.168.200.7`

6. The Application Gateway instance answers the client.

- Source IP address: `AppGwPIP`
- Destination IP address: `ClientPIP`

Outbound flows from the VMs to the public internet go through Azure Firewall, which the UDR to `0.0.0.0/0` defines.

As a variation of this design, you can configure private DNAT in Azure Firewall so that the application workload sees the IP addresses of the Azure Firewall instances as the source, and no UDRs are required. The source IP address of the application clients is already preserved in the `X-Forwarded-For` HTTP header by Application Gateway. So if Azure Firewall applies DNAT to the traffic, no information is lost. For more information, see [Filter inbound internet or intranet traffic with Azure Firewall policy DNAT by using the Azure portal](#).

Azure Firewall in front of Application Gateway design

This design lets Azure Firewall filter and discard malicious traffic before it reaches Application Gateway. For example, it can apply features like threat intelligence-based filtering. Another benefit is that the application gets the same public IP address for both inbound and outbound traffic, regardless of protocol. There are three modes in which you can theoretically configure Azure Firewall:

- **Azure Firewall with DNAT rules:** Azure Firewall only swaps IP addresses at the IP address layer, but it doesn't process the payload. As a result, it doesn't change any of the HTTP headers.
- **Azure Firewall with application rules and TLS inspection disabled:** Azure Firewall can look at the SNI header in TLS, but it doesn't decrypt it. A new TCP connection is created from the firewall to the next hop. In this example, it's Application Gateway.
- **Azure Firewall with application rules and TLS inspection enabled:** Azure Firewall looks into the packet contents and decrypts them. It serves as an HTTP proxy and can set the HTTP headers `X-Forwarded-For` to preserve the IP address. However, it presents a self-generated certificate to the client. For internet-based applications, using a self-generated certificate isn't an option because a security warning is sent to the application clients from their browser.

In the first two options, which are the only valid options for internet-based applications, Azure Firewall applies SNAT to the incoming traffic without setting the `X-Forwarded-For` header. As a result, the application can't see the original IP address of the HTTP requests. For administrative tasks, like troubleshooting, you can obtain the actual client IP address for a specific connection by correlating it with the SNAT logs of Azure Firewall.

The benefits of this scenario are limited because, unless you use TLS inspection and present self-generated certificates to the clients, Azure Firewall only sees encrypted traffic that goes to Application Gateway. This scenario is typically only possible for internal applications. However, there might be scenarios where this design is preferred. One scenario is if another Web Application Firewall exists earlier in the network (for example, with [Azure Front Door](#)), which can capture the original source IP in the `X-Forwarded-For` HTTP header. You might also prefer this design if many public IP addresses are required because Application Gateway supports a single IP address.

HTTP(S) inbound flows from the public internet should target the public IP address of Azure Firewall. Azure Firewall will DNAT and SNAT the packets to the private IP address of Application Gateway. From other Azure virtual networks or on-premises networks, HTTP(S) traffic should be

sent to the Application Gateway private IP address and forwarded through Azure Firewall with UDRs. Standard virtual network routing ensures that return traffic from the Azure VMs goes back to Application Gateway and from Application Gateway to Azure Firewall if DNAT rules were used. For traffic from on-premises or Azure, use UDRs in the Application Gateway subnet. For more information, see the packet walk later in this article. All outbound traffic from the Azure VMs to the internet is sent through Azure Firewall by UDRs.

The following table summarizes the traffic flows for this scenario.

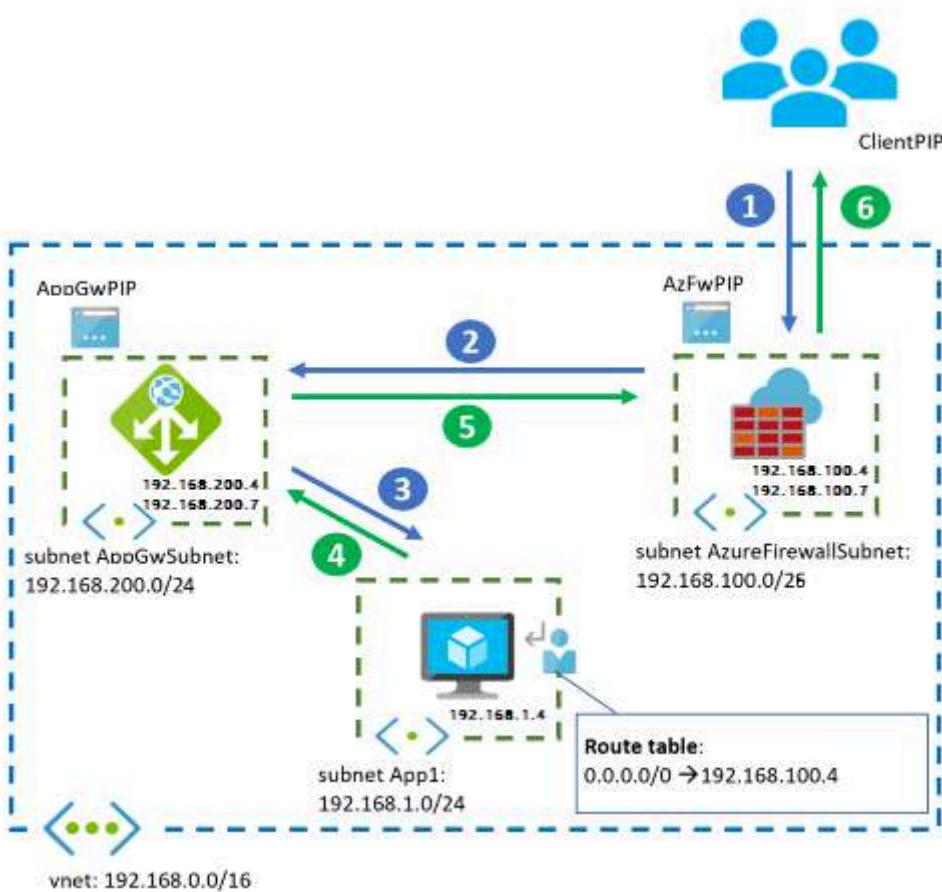
 [Expand table](#)

Flow	Goes through Application Gateway/Web Application Firewall	Goes through Azure Firewall
HTTP(S) traffic from the internet or on-premises to Azure	Yes	Yes
HTTP(S) traffic from Azure to the internet or on-premises	No	Yes
Non-HTTP(S) traffic from the internet or on-premises to Azure	No	Yes
Non-HTTP(S) traffic from Azure to the internet or on-premises	No	Yes

For inbound HTTP(S) traffic, Azure Firewall doesn't typically decrypt traffic. It instead applies IDPS policies that don't require TLS inspection, like IP address-based filtering or using HTTP headers.

Architecture

The application can't see the original source IP address of the web traffic. Azure Firewall applies SNAT to the packets as they come in to the virtual network. To avoid this problem, use [Azure Front Door](#) in front of the firewall. Azure Front Door injects the client's IP address as an HTTP header before it enters the Azure virtual network.



Workflow

Network traffic from the public internet follows this flow:

1. The client starts the connection to the public IP address of Azure Firewall.
 - Source IP address: **ClientPIP**
 - Destination IP address: **AzFwPIP**
2. The request to the Azure Firewall public IP address is distributed to a back-end instance of the firewall, which is **192.168.100.7** in this example. Azure Firewall applies DNAT to the web port, usually TCP 443, to the private IP address of the Application Gateway instance. Azure Firewall also applies SNAT when you perform DNAT. For more information, see [Azure Firewall known issues](#).
 - Source IP address, which is the private IP address of the Azure Firewall instance: **192.168.100.7**
 - Destination IP address: **192.168.200.4**
3. Application Gateway establishes a new session between the instance that handles the connection and one of the back-end servers. The original IP address of the client isn't in the packet.

- Source IP address, which is the private IP address of the Application Gateway instance: 192.168.200.7
- Destination IP address: 192.168.1.4
- X-Forwarded-For header: 192.168.100.7

4. The VM answers Application Gateway, which reverses both the source and destination IP addresses:

- Source IP address: 192.168.1.4
- Destination IP address: 192.168.200.7

5. Application Gateway replies to the SNAT source IP address of the Azure Firewall instance. Azure Firewall sees the internal IP address of Application Gateway, .4, as the source IP address, even if the connection comes from a specific Application Gateway instance like .7.

- Source IP address: 192.168.200.4
- Destination IP address: 192.168.100.7

6. Azure Firewall undoes SNAT and DNAT and answers the client.

- Source IP address: AzFwPIP
- Destination IP address: ClientPIP

Application Gateway needs a public IP address so that Microsoft can manage it, even if it has no listeners configured for applications.

! Note

A default route to 0.0.0.0/0 in the Application Gateway subnet that points to Azure Firewall isn't supported because it breaks the control plane traffic that Application Gateway requires to function properly.

On-premises clients

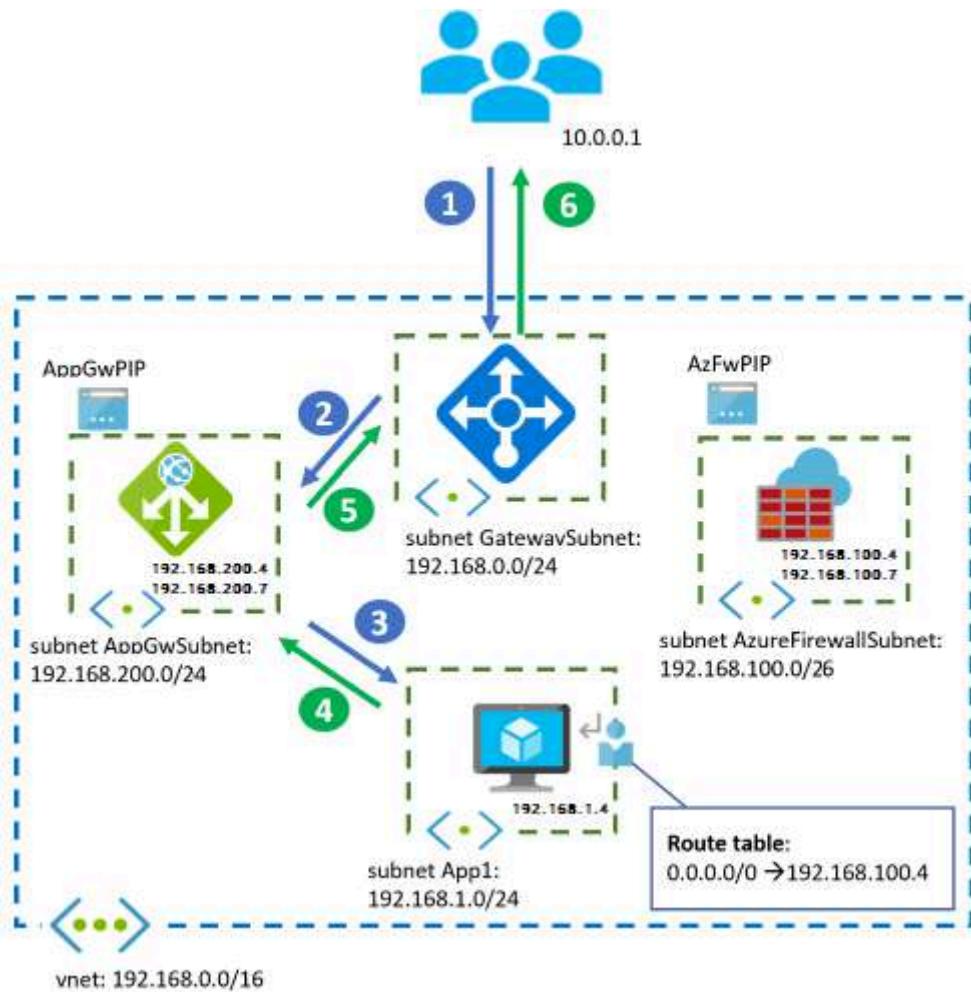
The preceding designs all show incoming application clients from the public internet. On-premises networks also access applications. Most of the previous information and traffic flows are the same as for internet clients, but there are some notable differences:

- A VPN gateway or ExpressRoute gateway sits in front of Azure Firewall or Application Gateway.

- Web Application Firewall uses the private IP address of Application Gateway.
- Azure Firewall doesn't support DNAT for private IP addresses, so you must use UDRs to send inbound traffic to Azure Firewall from the VPN or ExpressRoute gateways.
- Make sure to verify caveats around *forced tunneling* for [Application Gateway](#) and for [Azure Firewall](#). Even if your workload doesn't need outbound connectivity to the public internet, you can't inject a default route like `0.0.0.0/0` for Application Gateway that points to the on-premises network because it breaks control traffic. For Application Gateway, the default route needs to point to the public internet.

Architecture

The following diagram shows the Application Gateway and Azure Firewall parallel design. Application clients come from an on-premises network that's connected to Azure over VPN or ExpressRoute:

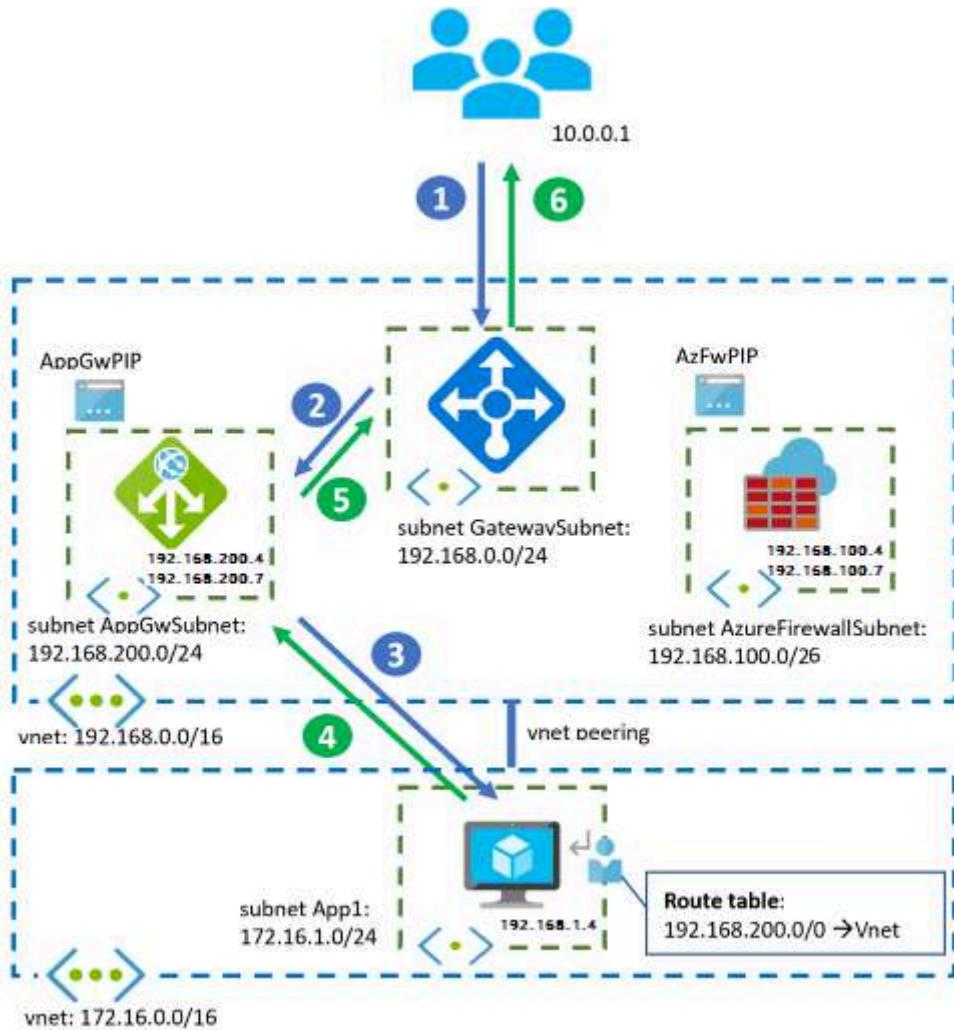


Even if all clients are located on-premises or in Azure, Application Gateway and Azure Firewall both need to have public IP addresses. These public IP addresses allow Microsoft to manage the services.

Hub-and-spoke topology

The designs in this article apply to a *hub-and-spoke* topology. Shared resources in a central hub virtual network connect to applications in separate spoke virtual networks through virtual network peerings.

Architecture



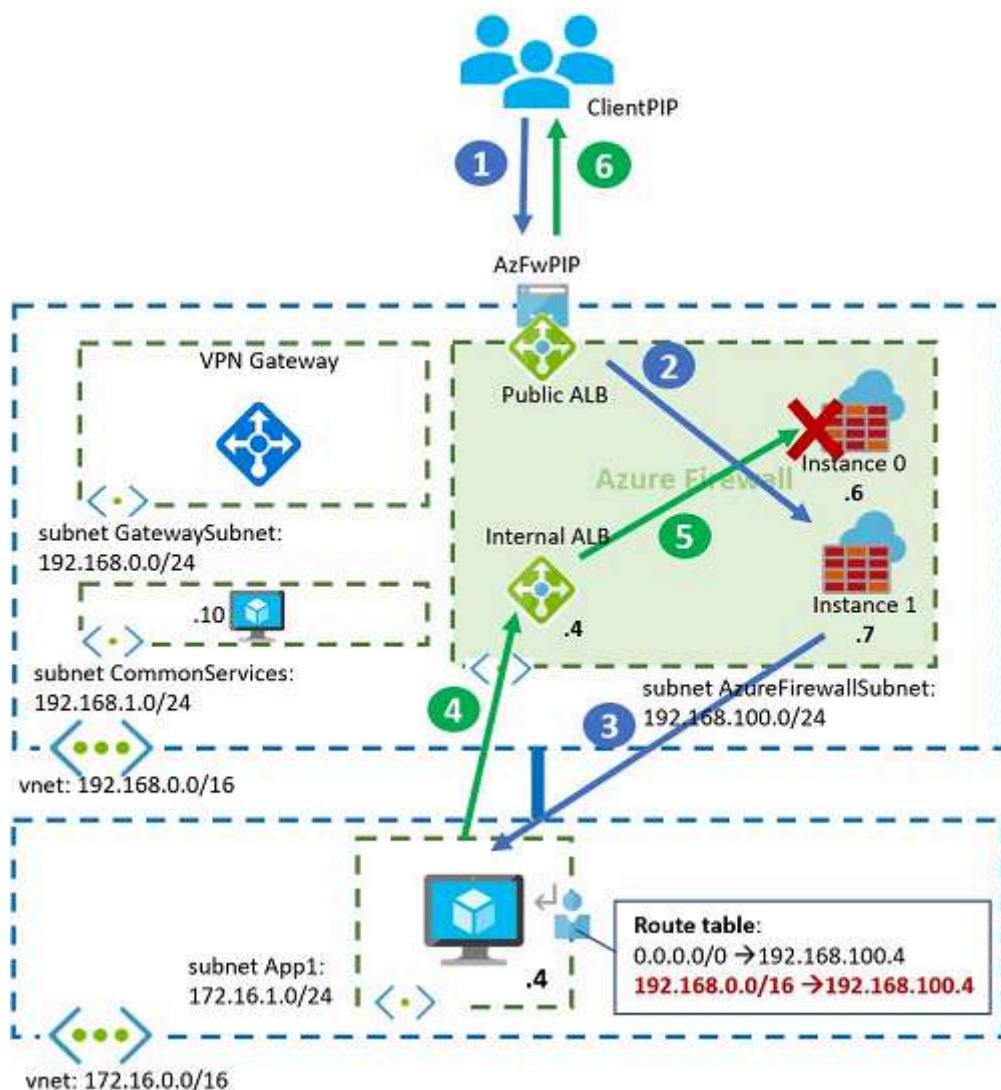
- Azure Firewall is deployed in the central hub virtual network. The previous diagram shows how to deploy Application Gateway in the hub. Application teams often manage components such as Application Gateways or API Management gateways. In this scenario, these components are deployed in the spoke virtual networks.
- Pay special attention to UDRs in the spoke networks. When an application server in a spoke receives traffic from a specific Azure Firewall instance, like the 192.168.100.7 IP address in the previous examples, it should send return traffic back to the same instance. If a UDR in the spoke sets the next hop of traffic addressed to the hub to the Azure Firewall IP address (192.168.100.4 in the previous diagrams), return packets might end up on a different Azure Firewall instance. This situation causes asymmetric routing. If you have UDRs in the spoke virtual networks, make sure to send traffic to shared services in

the hub through Azure Firewall. These UDRs don't include the prefix of the Azure Firewall subnet.

- The previous recommendation applies equally to the Application Gateway subnet and any other NVAs or reverse proxies that might be deployed in the hub virtual network.
- You can't set the next hop for the Application Gateway or Azure Firewall subnets through static routes with a next hop type of `Virtual Network`. This next hop type is only valid in the local virtual network and not across virtual network peerings. For more information about UDRs and next hop types, see [Virtual network traffic routing](#).

Asymmetric routing

The following diagram shows how a spoke sends SNAT traffic back to the Azure load balancer of Azure Firewall. This setup causes asymmetric routing.



To solve this problem, define UDRs in the spoke without the Azure Firewall subnet and with only the subnets where the shared services are located. In the previous diagram, the correct

UDR in the spoke should only contain `192.168.1.0/24`. It shouldn't contain the entire range `192.168.0.0/16`, which is marked in red.

Integration with other Azure products

You can integrate Azure Firewall and Application Gateway with other Azure products and services.

API Management Gateway

Integrate reverse proxy services like [API Management](#) gateway into the previous designs to provide functionality like API throttling or authentication proxy. API Management gateway integration doesn't significantly affect the designs. The key difference is that instead of the single Application Gateway reverse proxy, there are two reverse proxies chained behind each other.

For more information, see the [design guide to integrate API Management and Application Gateway in a virtual network](#) and the application pattern [API gateways for microservices](#).

AKS

For workloads that run on an AKS cluster, you can deploy Application Gateway independently of the cluster. Or you can integrate it with the AKS cluster by using the [Application Gateway Ingress Controller](#). When you configure specific objects at the Kubernetes levels, such as services and ingresses, Application Gateway automatically adapts without needing extra manual steps.

Azure Firewall plays an important role in AKS cluster security. It provides the required functionality to filter egress traffic from the AKS cluster based on FQDN, not only the IP address. For more information, see [Limit network traffic with Azure Firewall in AKS](#).

When you combine Application Gateway and Azure Firewall to protect an AKS cluster, it's best to use the parallel design option. Application Gateway with Web Application Firewall processes inbound connection requests to web applications in the cluster. Azure Firewall permits only explicitly allowed outbound connections. For more information about the parallel design option, see [Baseline architecture for an AKS cluster](#).

Azure Front Door

[Azure Front Door](#) has functionality that overlaps with Application Gateway in several areas. Both services provide web application firewalling, SSL offloading, and URL-based routing.

However, a key difference is that while Application Gateway operates within a virtual network, Azure Front Door is a global, decentralized service.

You can sometimes simplify virtual network design by replacing Application Gateway with a decentralized Azure Front Door. Most designs described in this article still apply, except for the option to place Azure Firewall in front of Azure Front Door.

One scenario is to use Azure Firewall in front of Application Gateway in your virtual network. Application Gateway injects the `X-Forwarded-For` header with the firewall instance's IP address, not the client's IP address. A workaround is to use Azure Front Door in front of the firewall to inject the client's IP address as a `X-Forwarded-For` header before the traffic enters the virtual network and reaches Azure Firewall. You can also [secure your origin with Azure Private Link in Azure Front Door Premium](#).

For more information about the differences between the two services, or when to use each one, see [Frequently asked questions for Azure Front Door](#).

Other NVAs

Microsoft products aren't the only choice to implement web application firewall or next-generation firewall functionality in Azure. A wide range of Microsoft partners provide NVAs. The concepts and designs are essentially the same as in this article, but there are some important considerations:

- Partner NVAs for next-generation firewalling might provide more control and flexibility for NAT configurations that Azure Firewall doesn't support. Examples include DNAT from on-premises or DNAT from the internet without SNAT.
- Azure-managed NVAs like Application Gateway and Azure Firewall reduce complexity, compared to NVAs where users need to handle scalability and resiliency across many appliances.
- When you use NVAs in Azure, use *active-active* and *autoscaling* setups so that these appliances aren't a bottleneck for applications that run in the virtual network.

Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal author:

- [Jose Moreno](#) | Principal Customer Engineer

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

Learn more about the component technologies:

- [What is Application Gateway?](#)
- [What is Azure Firewall?](#)
- [What is Azure Front Door?](#)
- [AKS](#)
- [What is Virtual Network?](#)
- [What is Web Application Firewall?](#)
- [Baseline architecture for an AKS cluster](#)

Related resources

Explore related architectures:

- [Implement a secure hybrid network](#)
- [Securely managed web applications](#)
- [High availability enterprise deployment by using App Service Environment](#)
- [Enterprise deployment by using App Service Environment](#)

Implement a Zero Trust network for web applications by using Azure Firewall and Azure Application Gateway

07/17/2025

This article describes how to implement Zero Trust security for web apps to enable inspection and end-to-end encryption. The Zero Trust model includes many other concepts, such as continuous identity verification and minimizing the size of the implicit trust areas.

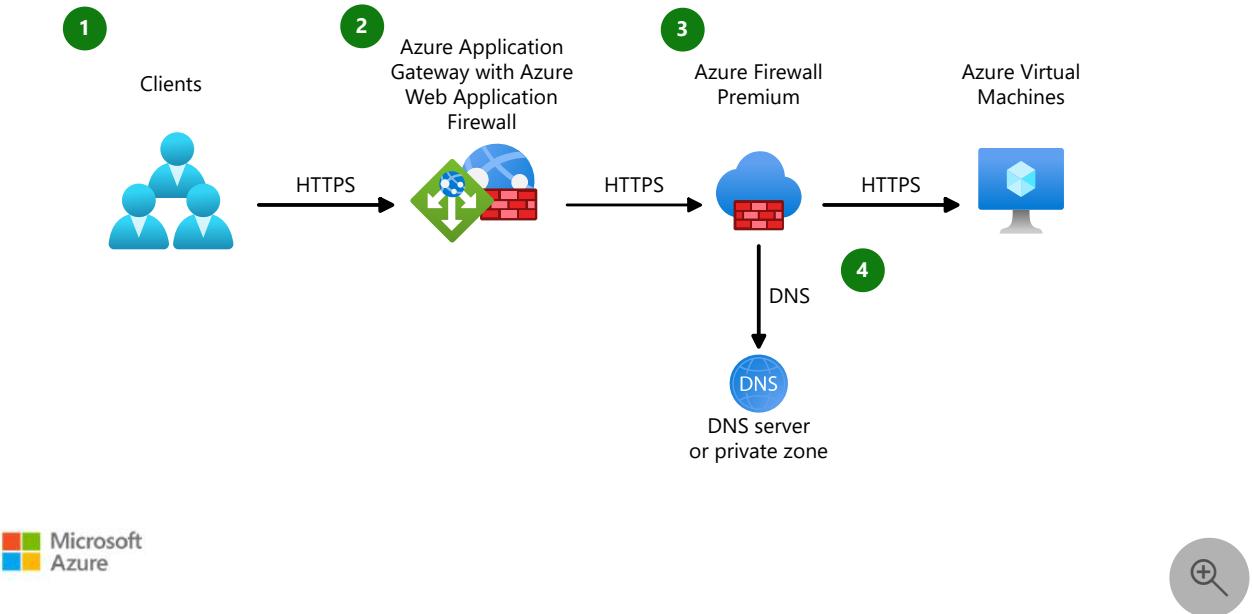
This article focuses on the encryption and inspection component of a Zero Trust architecture for inbound traffic from the public internet. For information about other aspects of deploying your application securely, such as authentication and authorization, see the [Zero Trust documentation](#). The example in this article uses a multilayered approach. In a multilayered approach, network security makes up one of the layers of the Zero Trust model. In this layer, network appliances inspect packets to ensure that only legitimate traffic reaches applications.

Typically, different types of network appliances inspect different aspects of network packets:

- Web application firewalls look for patterns that indicate an attack at the web application layer.
- Next-generation firewalls can also look for generic threats.

This architecture focuses on a common pattern for maximizing security, in which Azure Application Gateway inspects and processes traffic before it reaches Azure Firewall Premium. In some scenarios, you can combine different types of network security appliances to increase protection. For more information, see [Azure Firewall and Application Gateway for virtual networks](#).

Architecture



Download a [Visio file](#) of this architecture.

This architecture uses the Transport Layer Security (TLS) protocol to encrypt traffic at every step.

1. A client sends packets to Application Gateway, a load balancer. It runs with the optional addition of [Azure Web Application Firewall](#).
2. Application Gateway decrypts the packets and searches for threats to web applications. If it doesn't find any threats, it uses Zero Trust principles to encrypt the packets. Then it releases them.
3. Azure Firewall Premium runs the following security checks:
 - [TLS inspection](#) decrypts and examines the packets.
 - [Intrusion detection and prevention system \(IDPS\)](#) features check the packets for malicious intent.
4. If the packets pass the tests, Azure Firewall Premium takes these steps:
 - It encrypts the packets.
 - It uses a Domain Name System (DNS) service to determine the application virtual machine (VM).
 - It forwards the packets to the application VM.

Various inspection engines in this architecture ensure traffic integrity:

- Azure Web Application Firewall uses rules to prevent attacks at the web layer. Examples of attacks include SQL code injection and cross-site scripting. For more information about rules and the Open Worldwide Application Security Project (OWASP) Core Rule Set (CRS), see [Web application firewall CRS rule groups and rules](#).
- Azure Firewall Premium uses generic intrusion detection and prevention rules. These rules help identify malicious files and other threats that target web applications.

This architecture supports the following types of network design, which this article discusses:

- Traditional hub and spoke networks
- Networks that use Azure Virtual WAN as a platform
- Networks that use Azure Route Server to simplify dynamic routing

Azure Firewall Premium and name resolution

When Azure Firewall Premium checks for malicious traffic, it verifies that the HTTP Host header matches the packet IP address and Transmission Control Protocol (TCP) port. For example, suppose Application Gateway sends web packets to the IP address 172.16.1.4 and TCP port 443. The value of the HTTP Host header should resolve to that IP address.

HTTP Host headers usually don't contain IP addresses. Instead, the headers contain names that match the server's digital certificate. In this case, Azure Firewall Premium uses DNS to resolve the Host header name to an IP address. The network design determines which DNS solution works best.

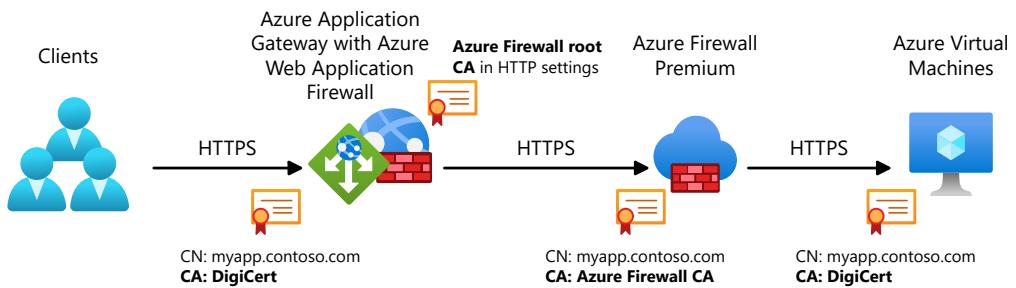
(!) Note

Application Gateway doesn't support port numbers in HTTP Host headers. As a result:

- Azure Firewall Premium assumes a default HTTPS TCP port of 443.
- The connection between Application Gateway and the web server only supports TCP port 443, not nonstandard ports.

Digital certificates

The following diagram shows the common names (CNs) and certificate authorities (CAs) that this architecture's TLS sessions and certificates use.



Azure Firewall dynamically generates its own certificates. This capability is one of the main reasons why it's placed behind Application Gateway. Otherwise, the application client is confronted with self-generated certificates that are flagged as a security risk.

TLS connections

This architecture contains three distinct TLS connections. Digital certificates validate each one.

From clients to Application Gateway

In Application Gateway, you deploy the digital certificate that clients see. A well-known CA such as DigiCert or Let's Encrypt typically issues such a certificate. This mechanism is fundamentally different from how Azure Firewall dynamically generates digital certificates from a self-signed or internal public key infrastructure CA.

From Application Gateway to Azure Firewall Premium

To decrypt and inspect TLS traffic, Azure Firewall Premium dynamically generates certificates. Azure Firewall Premium also presents itself to Application Gateway as the web server. A private CA signs the certificates that Azure Firewall Premium generates. For more information, see [Azure Firewall Premium certificates](#). Application Gateway needs to validate those certificates. In the application's HTTP settings, you configure the root CA that Azure Firewall Premium uses.

From Azure Firewall Premium to the web server

Azure Firewall Premium establishes a TLS session with the destination web server. Azure Firewall Premium verifies that a well-known CA signs the web server TLS packets.

Component roles

Application Gateway and Azure Firewall Premium handle certificates differently from one another because their roles differ:

- Application Gateway is a *reverse web proxy*. It protects web servers from malicious clients by intercepting HTTP and HTTPS requests. You declare each protected server that's in the back-end pool of Application Gateway with its IP address or fully qualified domain name. Legitimate clients should be able to access each application. So you configure Application Gateway with a digital certificate that a public CA signs. Use a CA that any TLS client accepts.
- Azure Firewall Premium is a *forward web proxy* or, simply, a web proxy. It protects clients from malicious web servers by intercepting TLS calls from the protected clients. When a protected client makes an HTTP request, the forward web proxy impersonates the target web server by generating digital certificates and presenting them to the client. Azure Firewall Premium uses a private CA, which signs the dynamically generated certificates. You configure the protected clients to trust that private CA. In this architecture, Azure Firewall Premium protects requests from Application Gateway to the web server. Application Gateway trusts the private CA that Azure Firewall Premium uses.

Routing and traffic forwarding

Routing is slightly different depending on the topology of your network design. The following sections describe examples of hub and spoke, Virtual WAN, and Route Server topologies. All topologies have the following aspects in common:

- Application Gateway always serves as a proxy. Azure Firewall Premium also serves as a proxy when it's configured for TLS inspection. Application Gateway terminates the TLS sessions from clients, and new TLS sessions are built toward Azure Firewall. Azure Firewall receives and terminates the TLS sessions sourced from Application Gateway and builds new TLS sessions toward the workloads. This process affects the IDPS configuration of Azure Firewall Premium. For more information, see [IDPS and private IP addresses](#).
- The workload sees connections that come from the Azure Firewall subnet IP address. The original client IP address is preserved in the `X-Forwarded-For` HTTP header that Application Gateway inserts. Azure Firewall also supports injecting the source client IP

address in the `x-Forwarded-For` header. In this scenario, the source client IP address is the application gateway's IP address.

- Traffic from Application Gateway to the workload is typically sent to Azure Firewall by using Azure routing mechanisms. These mechanisms include user-defined routes (UDRs) configured in the Application Gateway subnet or routes that Virtual WAN or Route Server inject. Explicitly defining the Azure Firewall private IP address in the Application Gateway back-end pool is possible, but we don't recommend doing so because it removes some of the native functionality of Application Gateway, such as load balancing and session stickiness.

The following sections describe some of the most common topologies that you can use with Azure Firewall and Application Gateway.

Hub and spoke topology

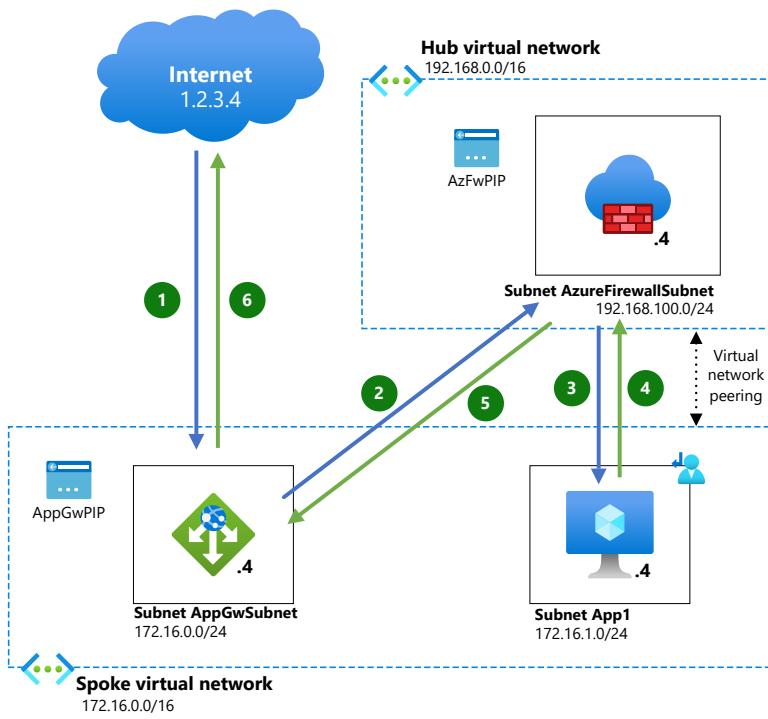
A hub and spoke design typically deploys shared network components in the hub virtual network and application-specific components in the spokes. In most systems, Azure Firewall Premium is a shared resource. Azure Web Application Firewall can be a shared network device or an application-specific component. It's a best practice to treat Application Gateway as an application component and deploy it in a spoke virtual network for the following reasons:

- It can be difficult to troubleshoot Azure Web Application Firewall alerts. You generally need in-depth knowledge of the application to decide whether the messages that trigger those alarms are legitimate.
- If you treat Application Gateway as a shared resource, you might exceed [Application Gateway limits](#).
- You might face role-based access control problems if you deploy Application Gateway in the hub. This situation can occur when teams manage different applications but use the same instance of Application Gateway. Each team then has access to the entire Application Gateway configuration.

In traditional hub and spoke architectures, DNS private zones provide an easy way to use DNS:

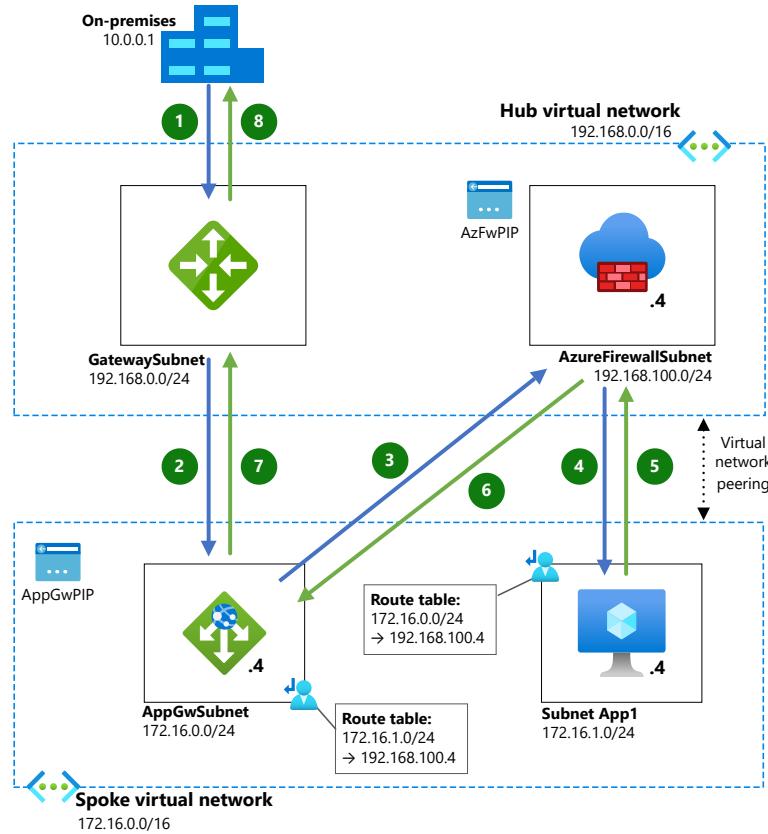
1. Configure a DNS private zone.
2. Link the zone to the virtual network that contains Azure Firewall Premium.
3. Make sure that an address record exists for the value that Application Gateway uses for traffic and for health checks.

The following diagram shows the packet flow when Application Gateway is in a spoke virtual network. In this case, a client connects from the public internet.



1. A client submits a request to a web server.
2. Application Gateway intercepts the client packets and examines them. If the packets pass inspection, Application Gateway sends the packets to the back-end VM. When the packets reach Azure, a UDR in the Application Gateway subnet forwards them to Azure Firewall Premium.
3. Azure Firewall Premium runs security checks on the packets. If they pass the tests, Azure Firewall Premium forwards the packets to the application VM.
4. The VM responds and sets the destination IP address to the application gateway. A UDR in the VM subnet redirects the packets to Azure Firewall Premium.
5. Azure Firewall Premium forwards the packets to Application Gateway.
6. Application Gateway answers the client.

Traffic can also arrive from an on-premises network instead of the public internet. The traffic flows either through a site-to-site virtual private network (VPN) or through Azure ExpressRoute. In this scenario, the traffic first reaches a virtual network gateway in the hub. The rest of the network flow is the same as the previous diagram.



1. An on-premises client connects to the virtual network gateway.
2. The virtual network gateway forwards the client packets to Application Gateway.
3. Application Gateway examines the packets. If they pass inspection, a UDR in the Application Gateway subnet forwards the packets to Azure Firewall Premium.
4. Azure Firewall Premium runs security checks on the packets. If they pass the tests, Azure Firewall Premium forwards the packets to the application VM.
5. The VM responds and sets the destination IP address to Application Gateway. A UDR in the VM subnet redirects the packets to Azure Firewall Premium.
6. Azure Firewall Premium forwards the packets to Application Gateway.
7. Application Gateway sends the packets to the virtual network gateway.
8. The virtual network gateway answers the client.

Virtual WAN topology

You can also use the networking service [Virtual WAN](#) in this architecture. This component provides many benefits. For instance, it eliminates the need for user-maintained UDRs in spoke virtual networks. You can define static routes in virtual hub route tables instead. The programming of every virtual network that you connect to the hub then contains these routes.

When you use Virtual WAN as a networking platform, two main differences result:

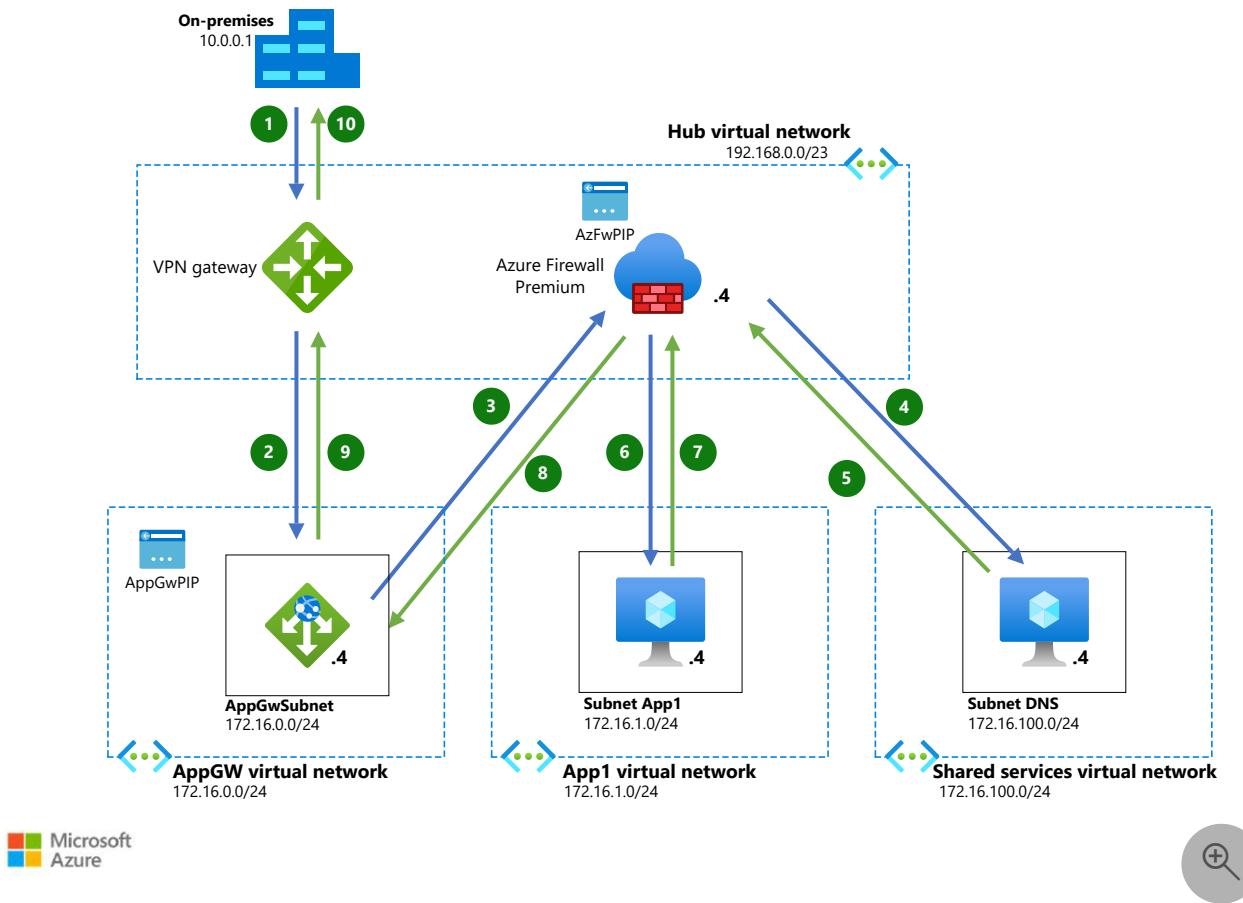
- You can't link DNS private zones to a virtual hub because Microsoft manages virtual hubs. As the subscription owner, you don't have permissions to link private DNS zones. As a result, you can't associate a DNS private zone with the secure hub that contains Azure Firewall Premium.

To implement DNS resolution for Azure Firewall Premium, use DNS servers instead:

- Configure the [Azure Firewall DNS settings](#) to use custom DNS servers.
- Deploy the servers in a shared services virtual network that you connect to the virtual WAN.
- Link a DNS private zone to the shared services virtual network. The DNS servers can then resolve the names that Application Gateway uses in HTTP Host headers. For more information, see [Azure Firewall DNS settings](#).
- You can only use Virtual WAN to program routes in a spoke if the prefix is shorter (less specific) than the virtual network prefix. For example, in the preceding diagrams, the spoke virtual network has the prefix `172.16.0.0/16`. In this case, Virtual WAN isn't able to inject a route that matches the virtual network prefix (`172.16.0.0/16`) or any of the subnets (`172.16.0.0/24`, `172.16.1.0/24`). In other words, Virtual WAN can't direct traffic between two subnets that are in the same virtual network.

This limitation becomes apparent when Application Gateway and the destination web server are in the same virtual network. Virtual WAN can't force the traffic between Application Gateway and the web server to go through Azure Firewall Premium. One work-around is to manually configure UDRs in the Application Gateway and web server subnets.

The following diagram shows the packet flow in an architecture that uses Virtual WAN. In this scenario, access to Application Gateway comes from an on-premises network. A site-to-site VPN or ExpressRoute instance connects that network to Virtual WAN. Internet-based access follows a similar path.



1. An on-premises client connects to the VPN gateway.
2. The VPN gateway forwards the client packets to Application Gateway.
3. Application Gateway examines the packets. If they pass inspection, the Application Gateway subnet forwards the packets to Azure Firewall Premium.
4. Azure Firewall Premium requests DNS resolution from a DNS server in the shared services virtual network.
5. The DNS server answers the resolution request.
6. Azure Firewall Premium runs security checks on the packets. If they pass the tests, Azure Firewall Premium forwards the packets to the application VM.
7. The VM responds and sets the destination IP address to Application Gateway. The application subnet redirects the packets to Azure Firewall Premium.
8. Azure Firewall Premium forwards the packets to Application Gateway.
9. Application Gateway sends the packets to the VPN gateway.
10. The VPN gateway answers the client.

If you use this design, you might need to modify the routing that the hub advertises to the spoke virtual networks. Specifically, Application Gateway v2 only supports a `0.0.0.0/0` route that points to the internet. Routes with this address that don't point to the internet break the connectivity that Microsoft requires for managing Application Gateway. If your virtual hub advertises a `0.0.0.0/0` route, prevent that route from propagating to the Application Gateway subnet by taking one of these steps:

- Create a route table with a route for `0.0.0.0/0` and a next hop type of `Internet`. Associate that route with the subnet that you deploy Application Gateway in.
- If you deploy Application Gateway in a dedicated spoke, disable the propagation of the default route in the settings for the virtual network connection.

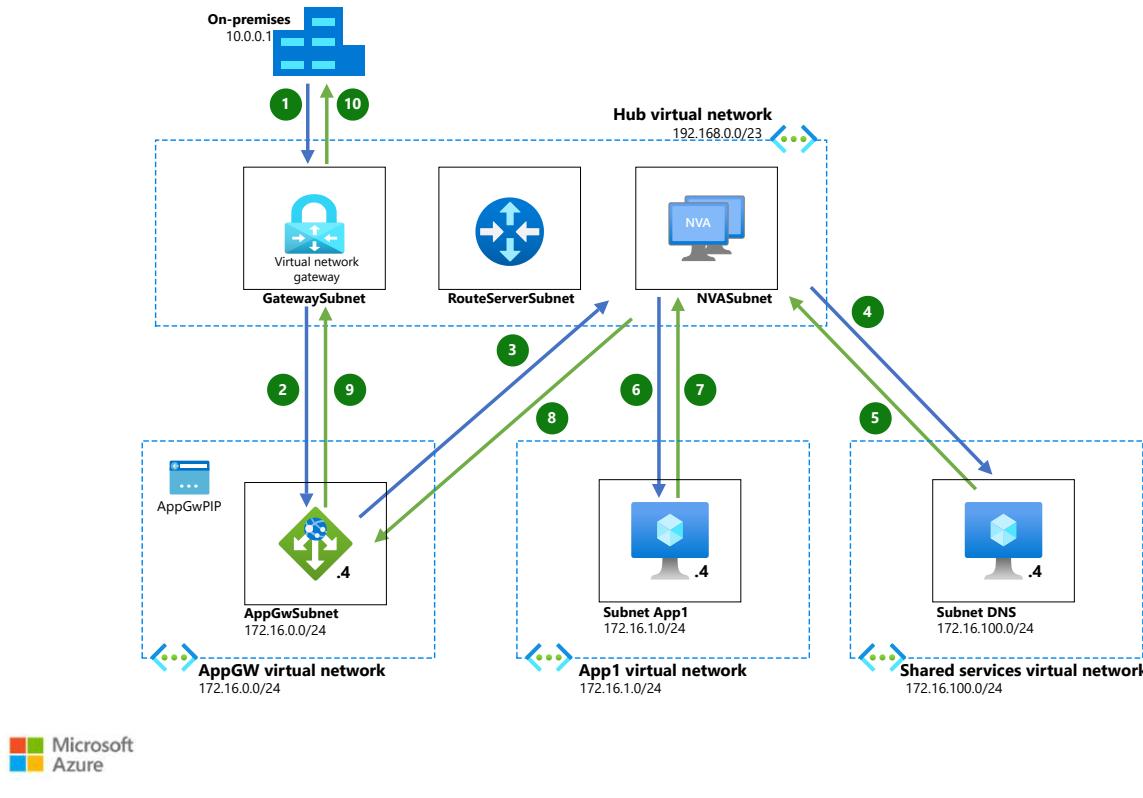
Route Server topology

[Route Server](#) provides another way to inject routes automatically in spokes. Use this functionality to avoid the administrative overhead of maintaining route tables. Route Server combines the Virtual WAN and hub and spoke variants:

- You can use Route Server to manage hub virtual networks. As a result, you can link the hub virtual network to a DNS private zone.
- Route Server has the same limitation that Virtual WAN has concerning IP address prefixes. You can only inject routes into a spoke if the prefix is shorter (less specific) than the virtual network prefix. Because of this limitation, Application Gateway and the destination web server need to be in different virtual networks.

The following diagram shows the packet flow when Route Server simplifies dynamic routing. Consider the following points:

- Route Server currently requires the device that injects the routes to send them over Border Gateway Protocol (BGP). Azure Firewall Premium doesn't support BGP, so use a non-Microsoft network virtual appliance (NVA) instead.
- The functionality of the NVA in the hub determines whether your implementation needs DNS.



1. An on-premises client connects to the virtual network gateway.
2. The virtual network gateway forwards the client packets to Application Gateway.
3. Application Gateway examines the packets. If they pass inspection, the Application Gateway subnet forwards the packets to a back-end machine. Route Server injects a route in the Application Gateway subnet that forwards the traffic to an NVA.
4. The NVA subnet requests DNS resolution from a DNS server in the shared services virtual network.
5. The DNS server answers the resolution request.
6. The NVA runs security checks on the packets. If they pass the tests, the NVA forwards the packets to the application VM.
7. The application VM responds and sets the destination IP address to Application Gateway. Route Server injects a route in the VM subnet that redirects the packets to the NVA.
8. The NVA forwards the packets to Application Gateway.
9. Application Gateway sends the packets to the virtual network gateway.
10. The virtual network gateway answers the client.

Like with Virtual WAN, you might need to modify the routing when you use Route Server. If you advertise the `0.0.0.0/0` route, it might propagate to the Application Gateway subnet. But Application Gateway doesn't support that route. In this case, configure a route table for the Application Gateway subnet. Include a route for `0.0.0.0/0` and a next hop type of `Internet` in that table.

IDPS and private IP addresses

Azure Firewall Premium decides which IDPS rules to apply based on the source and destination IP addresses of the packets. By default, Azure Firewall treats private IP addresses in the RFC 1918 ranges (`10.0.0.0/8`, `192.168.0.0/16`, and `172.16.0.0/12`) and RFC 6598 range (`100.64.0.0/10`) as internal. So, if you deploy Application Gateway in a subnet in one of these ranges, Azure Firewall Premium considers traffic between Application Gateway and the workload to be internal. Therefore, only IDPS signatures marked to be applied to internal traffic or to any traffic are used. IDPS signatures marked to be applied for inbound or outbound traffic aren't applied to traffic between Application Gateway and the workload. For more information, see [Azure Firewall IDPS rules](#).

The easiest way to force IDPS inbound signature rules to be applied to the traffic between Application Gateway and the workload is by placing Application Gateway in a subnet that uses a prefix outside of the private ranges. You don't necessarily need to use public IP addresses for this subnet. Instead, you can customize the IP addresses that Azure Firewall Premium treats as internal for IDPS. For example, if your organization doesn't use the `100.64.0.0/10` range, you can eliminate this range from the list of internal prefixes for IDPS and deploy Application Gateway in a subnet configured with an IP address in `100.64.0.0/10`. For more information, see [Azure Firewall Premium private IPDS ranges](#).

Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal author:

- [Jose Moreno](#) | Principal Customer Engineer

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

- [Secure networks with Zero Trust](#)
- [Virtual network traffic routing](#)

- How an application gateway works

Related resources

- [Implement a secure hybrid network](#)
- [Hub-spoke network topology in Azure](#)
- [Hub-spoke network topology that uses Virtual WAN](#)

Design a secure research environment for regulated data

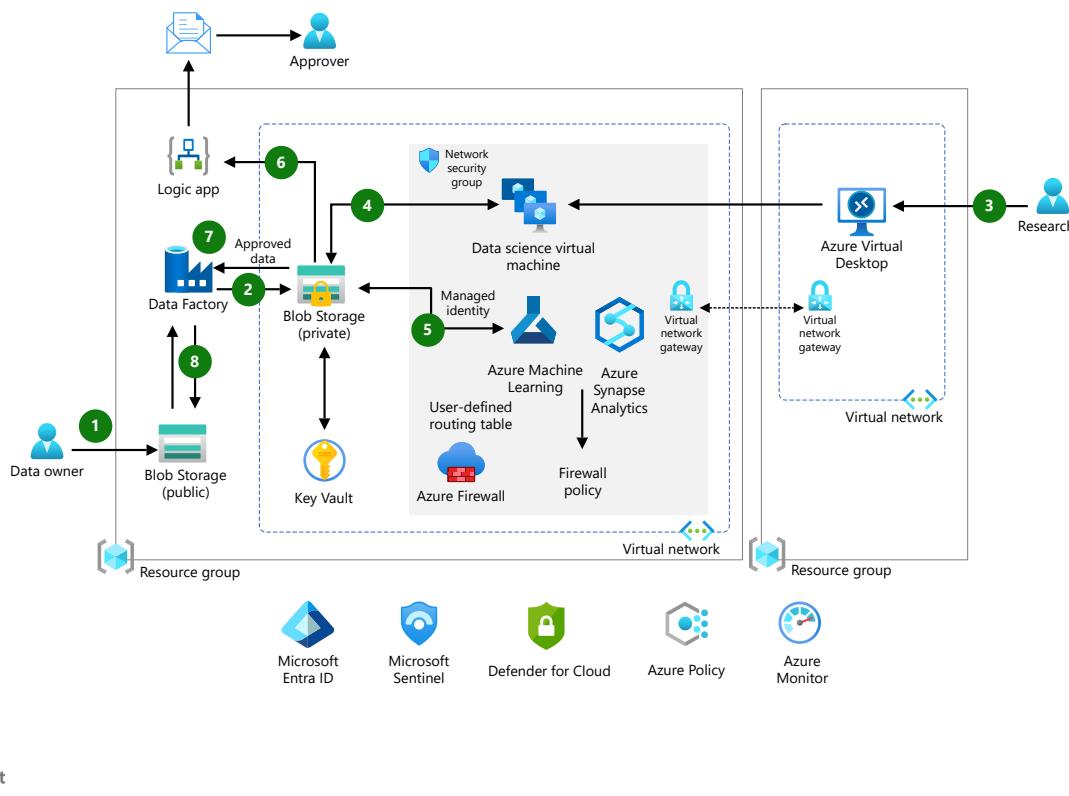
Azure Data Science Virtual Machines

Azure Machine Learning

Azure Data Factory

This article describes a secure research environment that allows researchers to access sensitive data that's under a high level of control and protection. This article applies to organizations that must adhere to regulatory compliance or other strict security requirements.

Architecture



Download a [Visio file](#) of this architecture.

Dataflow

The following or dataflow corresponds to the above diagram:

1. Data owners upload datasets into a public blob storage account. They use Microsoft-managed keys to encrypt the data.
2. **Azure Data Factory** uses a trigger that starts copying the uploaded dataset to a specific location, or import path, on another storage account that has security controls. You can only reach the storage account through a private endpoint. A service principal that has

limited permissions can also access the account. Data Factory deletes the original copy, which makes the dataset immutable.

3. Researchers access the secure environment through a streaming application by using [Azure Virtual Desktop](#) as a privileged jump box.
4. The dataset in the secure storage account is presented to the data science virtual machines (VMs) that you provision in a secure network environment for research work. Much of the data preparation is done on those VMs.
5. The secure environment has [Azure Machine Learning](#) and [Azure Synapse Analytics](#), which can access the dataset through a private endpoint. You can use these platforms to train, deploy, automate, and manage machine learning models or use Azure Synapse Analytics. At this point, you can create models that meet regulatory guidelines. De-identify all model data by removing personal information.
6. Models or de-identified data are saved to a separate location on the secure storage, or export path. When you add new data to the export path, you trigger a logic app. In this architecture, the logic app is outside of the secure environment because no data is sent to the logic app. Its only function is to send notifications and start the manual approval process.

The logic app starts an approval process by requesting a review of data that's queued to be exported. The manual reviewers help ensure that sensitive data isn't exported. After the review process, the data is either approved or denied.

 **Note**

If an approval step isn't required on exfiltration, you can omit the logic app step.

7. If the de-identified data is approved, it's sent to the Data Factory instance.
8. Data Factory moves the data to the public storage account in a separate container to allow external researchers to access their exported data and models. Alternately, you can provision another storage account in a lower security environment.

Components

This architecture consists of several Azure services that scale resources according to your needs. The following sections describe these services and their roles. For links to product documentation to get started with these services, see [Next steps](#).

Core workload components

Here are the core components that move and process research data.

- **Azure data science VMs** are VMs that you configure with tools for data analytics and machine learning. In this architecture, they provide researchers with dedicated, secure compute resources for data preparation, analysis, and model training within the isolated environment. Use the data science VM when you need specific packages or tools, such as MATLAB or SAS, that platform as a service (PaaS) environments can't support. For security and ease of use, choose Machine Learning and other PaaS options when they're supported.
- **Machine Learning** is a service that you can use to train, deploy, automate, and manage machine learning models. In this architecture, it facilitates model development and orchestration while maintaining security controls over data access and compute resources. You can also use it to manage the allocation and use of machine learning compute resources. Machine Learning is the tool of choice for Jupyter notebooks for development.
- **Machine Learning compute** is a cluster of nodes that you can use to train and test machine learning and AI models. In this architecture, it provides automatically scalable, secure, and isolated compute resources for research. You can deploy Visual Studio Code (VS Code) as a streaming application from Virtual Desktop and connect it to the Machine Learning compute for an alternative development environment.
- **Azure Blob Storage** is an object storage solution for storing unstructured data in the cloud. In this architecture, it's the primary storage solution, and it has two instances. The public instance temporarily stores the data that the data owners upload. The public instance also stores de-identified data after it models the data in a separate container. The second instance is private. It receives the training and test datasets from Machine Learning that the training scripts use. Storage is mounted as a virtual drive onto each node of a Machine Learning compute cluster.
- **Data Factory** is a managed cloud service that orchestrates and operationalizes processes to move raw data between systems. In this architecture, it moves data between storage accounts of differing security levels, enforces separation of duties, and manages data flows throughout the secure environment.
- **Azure Synapse Analytics** is an analytical tool for big data and pipelines for data integration and extract, transform, load workloads. Azure Synapse Analytics is also a preferred service to run Apache Spark workloads. In this architecture, it enables advanced analytics and data integration for research datasets that can be accessed through secure, private endpoints.

- **Virtual Desktop** is a desktop and app virtualization service that runs on the cloud. In this architecture, it acts as a jump box that you can use to gain access to the resources in the secure environment. It enables researchers to connect to data science VMs by using streaming applications and a full desktop, as needed.

Alternatively, you can use [Azure Bastion](#), but you should have a clear understanding of the security control differences between the two options. Virtual Desktop has some advantages, including:

- The ability to stream an app like VS Code to run notebooks on the machine learning compute resources.
 - The ability to limit copy, paste, and screen captures.
 - Support for Microsoft Entra authentication to data science VMs.
- **Azure Logic Apps** provides automated low-code workflows. In this architecture, it manages the *trigger* and *release* portions of the manual approval process.

Posture management components

These components continuously monitor the posture of the workload and its environment. Their purpose is to discover and mitigate risks as soon as they're discovered.

- **Microsoft Defender for Cloud** is a service that you can use to evaluate the overall security posture of the implementation and provide an attestation mechanism for regulatory compliance. In this architecture, it helps you discover problems early, instead of when you perform audits or assessments. Use features to track progress such as the secure score and compliance score. These scores are important tools that help verify compliance.
- **Microsoft Sentinel** is a security information and event management solution and a security orchestration, automation, and response solution. In this architecture, it centralizes logs, detects threats, and automates security responses for the research environment. You can centrally view logs and alerts from various sources and take advantage of advanced AI and security analytics to detect, hunt, prevent, and respond to threats. This capability provides valuable security insights to help ensure that traffic and any activities associated with the workspace meet your expectations.
- **Azure Monitor** provides observability across your entire environment. In this architecture, it collects and visualizes metrics, activity logs, and diagnostics to support operational monitoring and incident detection. Management tools, such as tools in Defender for Cloud, also push log data to Azure Monitor.

Governance components

- [Azure Policy](#) is a governance tool for enforcing organizational standards and assessing compliance at scale. In this architecture, it helps ensure that resources and workloads adhere to security and configuration policies.

Alternatives

- This solution uses Data Factory to move data to the public storage account in a separate container to allow external researchers to have access to their exported data and models. Alternatively, you can provision another storage account in a lower security environment.
- This solution uses Virtual Desktop as a jump box to gain access to the resources in the secure environment with streaming applications and a full desktop. Alternatively, you can use Azure Bastion, but Virtual Desktop has some advantages. These advantages include the ability to stream an app, to limit copy/paste and screen captures, and to support Microsoft Entra authentication. You can also consider configuring a Point-to-Site VPN for offline training locally. This VPN also helps reduce the cost of having multiple VMs for workstations.
- To secure data at rest, this solution encrypts all Azure Storage accounts with Microsoft-managed keys by using strong cryptography. Alternatively, you can use customer-managed keys. You must store the keys in a managed key store.

Scenario details

This scenario combines regulated and private data that individuals must access but aren't allowed to store or transmit.

- Data scientists outside of your organization need full access to the data to train and export their models without any proprietary or protected data leaving the environment.
- You must isolate access. Even the data owners and custodians aren't allowed to access the data after it's uploaded into the environment.
- You must require an audit trail for any exports that are transferred out of the environment to ensure that only the models were exported.

Potential use cases

This architecture was originally created for higher education research institutions with Health Insurance Portability and Accountability Act (HIPAA) requirements. However, you can use this design in any industry that requires the isolation of data for research purposes. Some examples include:

- Industries that process regulated data per National Institute of Standards and Technology (NIST) requirements.

- Medical centers that collaborate with internal or external researchers.
- Banking and finance industries.

By following the guidance in this article, you can maintain full control of your research data, have separation of duties, and meet strict regulatory compliance standards. This approach also facilitates collaboration among key roles in a research-oriented environment, such as data owners, researchers, and approvers.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that can be used to improve the quality of a workload. For more information, see [Microsoft Azure Well-Architected Framework](#).

Reliability

Reliability ensures your application can meet the commitments you make to your customers. For more information, see [Design review checklist for Reliability](#).

Most research solutions are temporary workloads and don't need to be available for extended periods. This architecture is designed as a single-region deployment with availability zones. If the business requirements demand higher availability, replicate this architecture in multiple regions. You need other components, such as a global load balancer and distributor, to route traffic to all those regions. As part of your recovery strategy, use Azure VM Image Builder to capture and create a copy of the customized base image.

Security

Security provides assurances against deliberate attacks and the abuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

The main objective of this architecture is to provide a secure and trusted research environment that strictly limits the exfiltration of data from the secure area.

Network security

Provision Azure resources that are used to store, test, and train research datasets in a secure environment. That environment is an Azure virtual network that has network security group rules to restrict access. These rules apply to:

- Inbound and outbound access to the public internet and within the virtual network.

- Access to and from specific services and ports. For example, this architecture blocks all port ranges except the ones required for Azure services, such as Azure Monitor. For a full list of service tags and the corresponding services, see [Virtual network service tags](#).

Access from the virtual network that has Virtual Desktop is restricted to approved access methods on specific ports, but all other traffic is denied. When compared to this environment, the other virtual network that has Virtual Desktop is relatively open.

The main blob storage in the secure environment is off the public internet. You can access it only within the virtual network through [private endpoint connections](#) and Storage firewalls. Use it to limit the networks from which clients can connect to file shares in Azure Files.

This architecture uses credential-based authentication for the main data store that's in the secure environment. In this case, the connection information, like the subscription ID and token authorization, is stored in a key vault. Another option is to create identity-based data access, where you use your Azure account to confirm whether you have access to Storage. In an identity-based data access scenario, no authentication credentials are saved. For more information, see [Create datastores](#).

The compute cluster can communicate only within the virtual network by using the Azure Private Link ecosystem and service or private endpoints, instead of using public IPs for communication. Make sure that you enable **No public IP**. For more information about this feature, which is currently in preview, see [Compute instance/cluster or serverless compute with no public IP](#).

The secure environment uses Machine Learning compute to access the dataset through a private endpoint. You can also configure Azure Firewall to control access to Machine Learning compute, which resides in a machine learning workspace. Use Azure Firewall to control outbound access from Machine Learning compute. For more information, see [Configure inbound and outbound network traffic](#).

To learn about one of the ways that you can secure a Machine Learning environment, see the blog post [Secure Machine Learning service environment](#).

For Azure services that you can't configure effectively with private endpoints, or to provide stateful packet inspection, consider using Azure Firewall or a non-Microsoft network virtual appliance.

Identity management

Access blob storage through Azure role-based access controls.

Virtual Desktop supports Microsoft Entra authentication to data science VMs.

Data Factory uses managed identity to access data from the blob storage. Data science VMs also use managed identity for remediation tasks.

Data security

To secure data at rest, all Storage accounts are encrypted with Microsoft-managed keys that use strong cryptography.

Alternatively, you can use customer-managed keys. You must store the keys in a managed key store. In this architecture, you deploy Azure Key Vault in the secure environment to store secrets like encryption keys and certificates. Resources in the secure virtual network access Key Vault through a private endpoint.

Governance considerations

Enable Azure Policy to enforce standards and provide automated remediation to bring resources into compliance for specific policies. You can apply the policies to a project subscription or at a management group level, either as a single policy or as part of a regulatory initiative.

For example, in this architecture, Azure machine configuration applies to all in-scope VMs. The policy can audit operating systems and machine configuration for the data science VMs.

VM image

The data science VMs run customized base images. To build the base image, use technologies like VM Image Builder. By using VM Image Builder, you can create a repeatable image that you can deploy when needed.

The base image might need updates, such as extra binaries. You should upload those binaries to the public blob storage. They should flow through the secure environment, much like how data owners upload the datasets.

Cost Optimization

Cost Optimization is about looking at ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

The cost of data science VMs depends on the choice of the underlying VM series. Because the workload is temporary, we recommend the consumption plan for the logic app resource. Use the [Azure pricing calculator](#) to estimate costs based on the estimated sizing of resources that

you need. Ensure that you shut down the environment when it's not in use to help optimize costs and improve security.

Performance Efficiency

Performance Efficiency is the ability of your workload to scale to meet the demands placed on it by users in an efficient manner. For more information, see [Design review checklist for Performance Efficiency](#).

The size and type of the data science VMs should be appropriate for the style of work that they do. This architecture is intended to support a single research project. You achieve scalability by adjusting the size and type of the VMs and by choosing compute resources that are available to Machine Learning.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Clayton Barlow ↗](#) | Senior Azure Specialist

Next steps

- [What is the data science VM for Linux and Windows?](#)
- [What is Machine Learning?](#)
- [What are compute targets in Machine Learning?](#)
- [Introduction to Blob Storage](#)
- [Introduction to Data Factory](#)
- [What is Virtual Desktop?](#)
- [Defender for Cloud documentation](#)
- [What is Microsoft Sentinel?](#)
- [Azure Monitor overview](#)
- [What is Azure Policy?](#)
- [Understand Azure machine configuration](#)

Related resources

- [Compare Microsoft machine learning products and technologies](#)
- [Many models machine learning at scale with Machine Learning](#)

Computer forensics chain of custody in Azure

Azure Automation

Azure Disk Encryption

Azure Key Vault

Azure Storage Accounts

This article outlines an infrastructure and workflow process designed to help teams provide digital evidence that demonstrates a valid chain of custody in response to legal requests. This article describes how to maintain a valid chain of custody throughout the stages of evidence acquisition, preservation, and access.

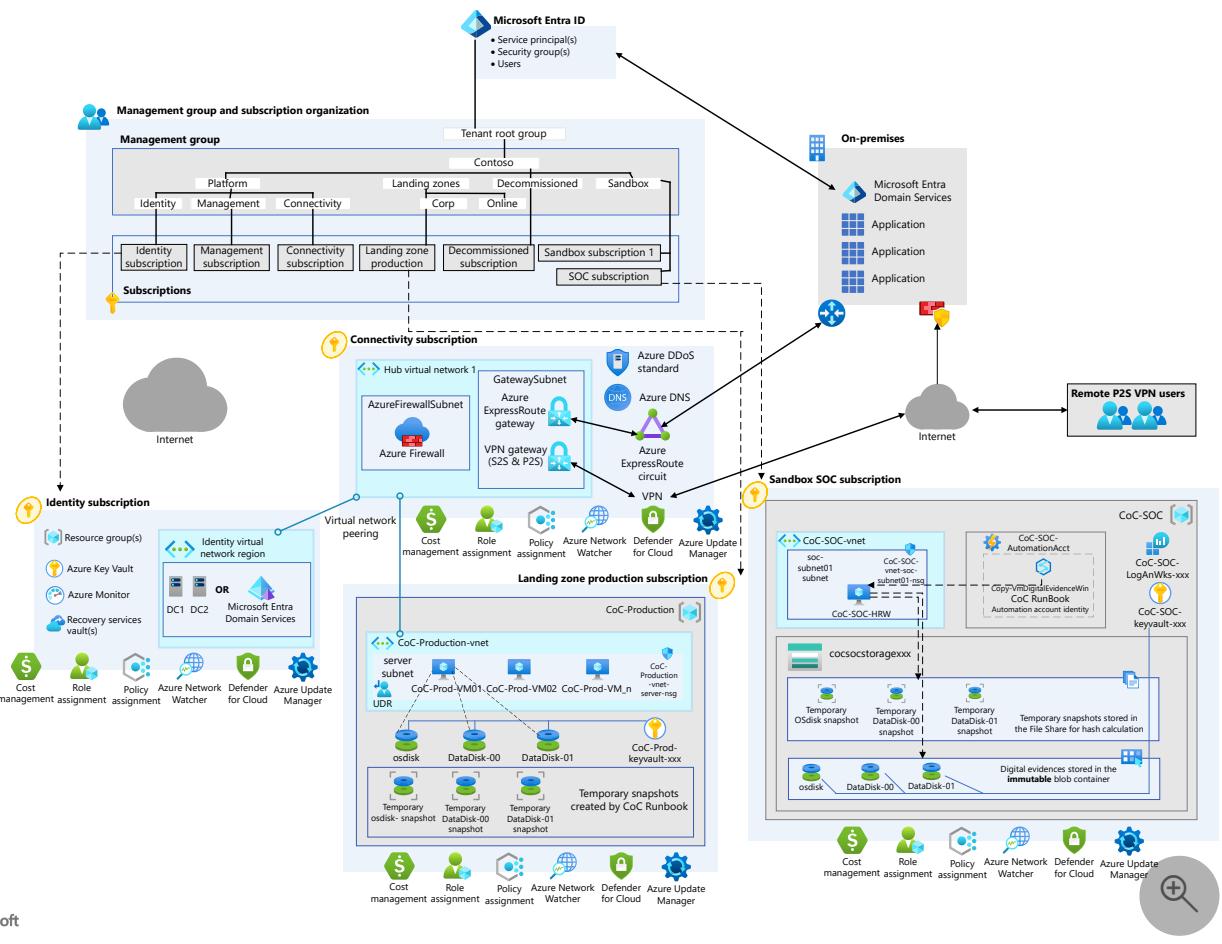
! Note

This article is based on the theoretical and practical knowledge of the authors. Before you use it for legal purposes, validate its applicability with your legal department.

Architecture

The architecture design follows the [Azure landing zone principles](#) in the Cloud Adoption Framework for Azure.

This scenario uses a hub-and-spoke network topology, which is shown in the following diagram:



Download a [Visio file](#) of this architecture.

Workflow

In the architecture, the production virtual machines (VMs) are part of a spoke [Azure virtual network](#). The VM disks are encrypted with Azure Disk Encryption. For more information, see [Overview of managed disk encryption options](#). In the production subscription, [Azure Key Vault](#) stores the BitLocker encryption keys (BEKs) of the VMs.

! Note

The scenario also supports production VMs that have unencrypted disks.

The security operations center (SOC) team uses a discrete Azure SOC subscription. The team has exclusive access to that subscription, which contains the resources that must be kept protected, inviolable, and monitored. The [Azure Storage](#) account in the SOC subscription hosts copies of disk snapshots in [immutable blob storage](#). A dedicated [key vault](#) stores copies of the hash values of the snapshots and the BEKs from the VMs.

In response to a request to capture the digital evidence of a VM, a member of the SOC team signs in to the Azure SOC subscription and uses an [Azure hybrid runbook worker](#) VM from

Azure Automation to run the `Copy-VmDigitalEvidence` runbook. The [Automation hybrid runbook worker](#) provides control of all mechanisms included in the capture.

The `Copy-VmDigitalEvidence` runbook implements the following macro steps:

1. Use the [system-assigned managed identity for an Automation account](#) to sign in to Azure. This identity grants access to the target VM's resources and the other Azure services needed for the solution.
2. Generate disk snapshots of the VM's operating system (OS) and data disks.
3. Transfer the snapshots to both the SOC subscription's immutable blob storage and a temporary file share.
4. Compute the hash values of the snapshots by using the copy that's stored in the file share.
5. Store the obtained hash values and the VM's BEK in the SOC key vault.
6. Remove all the copies of the snapshots, except for the copy in immutable blob storage.

 **Note**

The encrypted disks of the production VMs can also use key encryption keys (KEKs). The `Copy-VmDigitalEvidence` runbook provided in the [deploy scenario](#) doesn't cover this scenario.

Components

- [Azure Automation](#) is a cloud-based service that automates operational tasks by using runbooks and scripts. In this architecture, it orchestrates the evidence capture process by running the `Copy-VmDigitalEvidence` runbook to snapshot and transfer VM disks securely. This process helps ensure evidence integrity.
- [Azure Storage](#) is a scalable cloud storage solution for various data types, including object, file, disk, queue, and table storage. In this architecture, it stores VM disk snapshots in immutable blob containers to preserve digital evidence in a tamper-proof format.
- [Azure Blob Storage](#) is a cloud-based solution that provides object storage optimized for unstructured data. In this architecture, it holds the immutable snapshots of VM disks to ensure the integrity and non-repudiation of digital evidence.

- [Azure Files](#) is a fully managed cloud file storage service that provides shared file systems that can be accessed via the industry-standard Server Message Block (SMB) protocol, the Network File System (NFS) protocol, and the Azure Files REST API. You can concurrently mount shares through cloud or on-premises deployments of Windows, Linux, and macOS. You can also cache file shares on Windows Server by using Azure File Sync for quick access near the data usage location. In this architecture, Azure Files temporarily stores disk snapshots to compute hash values before transferring them to immutable storage.
- [Key Vault](#) is a secure cloud service for managing secrets, encryption keys, and certificates. In this architecture, it stores BEKs and hash values of disk snapshots to protect access and verify the integrity of digital evidence.
- [Microsoft Entra ID](#) is a cloud-based identity service that helps you control access to Azure and other cloud apps. In this architecture, it ensures that only authorized SOC personnel can access and manage sensitive evidence-handling operations.
- [Azure Monitor](#) is a monitoring service that provides observability through metrics, logs, and alerts. It supports operations at scale by helping you maximize the performance and availability of your resources, while proactively identifying potential problems. In this architecture, it archives activity logs to support auditing, compliance, and monitoring of the evidence chain of custody.

Automation

The SOC team uses an [Automation](#) account to create and maintain the `Copy-VmDigitalEvidence` runbook. The team also uses Automation to create the hybrid runbook workers that implement the runbook.

Hybrid runbook worker

The [hybrid runbook worker](#) VM is integrated into the Automation account. The SOC team uses this VM exclusively to run the `Copy-VmDigitalEvidence` runbook.

You must place the hybrid runbook worker VM in a subnet that can access the Storage account. Configure access to the Storage account by adding the hybrid runbook worker VM subnet to the Storage account's firewall allowlist rules.

Grant access to this VM only to the SOC team members for maintenance activities.

To isolate the virtual network that the VM uses, avoid connecting the virtual network to the hub.

The hybrid runbook worker uses the [Automation system-assigned managed identity](#) to access the target VM's resources and the other Azure services that the solution requires.

The minimum Azure role-based access control (Azure RBAC) permissions required for a system-assigned managed identity are divided into two categories:

- Access permissions to the SOC Azure architecture that contains the solution core components
- Access permissions to the target architecture that contains the target VM resources

Access to the SOC Azure architecture includes the following roles:

- **Storage Account Contributor** on the SOC immutable Storage account
- **Key Vault Secrets Officer** on the SOC key vault for BEK management

Access to the target architecture includes the following roles:

- **Contributor** on the target VM's resource group, which provides snapshot rights on VM disks
- **Key Vault Secrets Officer** on the target VM's key vault that's used to store the BEK, only if Azure RBAC is used to control the Key Vault access
- Access policy to **Get Secret** on the target VM's key vault that's used to store the BEK, only if the access policy is used to control the Key Vault access

Note

To read the BEK, the target VM's key vault must be accessible from the hybrid runbook worker VM. If the key vault's firewall is enabled, make sure that the public IP address of the hybrid runbook worker VM is permitted through the firewall.

Storage account

The [Storage account](#) in the SOC subscription hosts the disk snapshots in a container that's configured with a *legal hold* policy as Azure immutable blob storage. Immutable blob storage stores business-critical data objects in a write once, read many (WORM) state. The WORM state makes the data nonerasable and uneditable for a user-specified interval.

Make sure that you enable the [secure transfer](#) and [storage firewall](#) properties. The firewall grants access only from the SOC virtual network.

The storage account also hosts an [Azure file share](#) as a temporary repository that's used to calculate the snapshot's hash value.

Key Vault

The SOC subscription has its own instance of [Key Vault](#), which hosts a copy of the BEK that Azure Disk Encryption uses to protect the target VM. The primary copy is stored in the key vault that the target VM uses. This setup allows the target VM to continue normal operations without interruption.

The SOC key vault also stores the hash values of disk snapshots that the hybrid runbook worker computes during the capture operations.

Ensure that the [firewall](#) is enabled on the key vault. It must grant access exclusively from the SOC virtual network.

Log Analytics

A [Log Analytics workspace](#) stores activity logs used to audit all relevant events on the SOC subscription. Log Analytics is a feature of [Monitor](#).

Scenario details

Digital forensics is a science that addresses the recovery and investigation of digital data to support criminal investigations or civil proceedings. Computer forensics is a branch of digital forensics that captures and analyzes data from computers, VMs, and digital storage media.

Companies must guarantee that the digital evidence they provide in response to legal requests demonstrates a valid chain of custody throughout the stages of evidence acquisition, preservation, and access.

Potential use cases

- A company's SOC team can implement this technical solution to support a valid chain of custody for digital evidence.
- Investigators can attach disk copies that are obtained by using this technique on a computer that's dedicated to forensic analysis. They can attach the disk copies without powering on or accessing the original source VM.

Chain of custody regulatory compliance

If it's necessary to submit the proposed solution to a regulatory compliance validation process, consider the materials in the [considerations](#) section during the chain of custody solution validation process.

(!) Note

You should include your legal department in the validation process.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

The principles that validate this solution as a chain of custody are described in this section. To help ensure a valid chain of custody, digital evidence storage must demonstrate adequate access control, data protection and integrity, monitoring and alerting, and logging and auditing.

Security

Security provides assurances against deliberate attacks and the misuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

Compliance with security standards and regulations

When you validate a chain of custody solution, one of the requirements to evaluate is the compliance with security standards and regulations.

All the components included in the [architecture](#) are Azure standard services built on a foundation that supports trust, security, and [compliance](#).

Azure has a wide range of compliance certifications, including certifications tailored to countries or regions, and for key industries like healthcare, government, finance, and education.

For more information about updated audit reports that detail standards compliance for the services used in this solution, see [Service Trust Portal](#).

[Cohasset's Azure Storage compliance assessment](#) provides details about the following requirements:

- Securities and Exchange Commission (SEC) in 17 CFR § 240.17a-4(f), which regulates exchange members, brokers, or dealers.

- Financial Industry Regulatory Authority (FINRA) Rule 4511(c), which defers to the format and media requirements of SEC Rule 17a-4(f).
- Commodity Futures Trading Commission (CFTC) in regulation 17 CFR § 1.31(c)-(d), which regulates commodity futures trading.

It's Cohasset's opinion that Azure Storage, with the immutable storage feature of Blob Storage and policy lock option, retains time-based blobs (or *records*) in a nonerasable and nonrewriteable format and meets relevant storage requirements of SEC Rule 17a-4(f), FINRA Rule 4511(c), and the principles-based requirements of CFTC Rule 1.31(c)-(d).

Least privilege

When the roles of the SOC team are assigned, only two individuals in the team, known as SOC team custodians, should have rights to modify the [Azure RBAC](#) configuration of the subscription and its data. Grant other individuals only bare minimum access rights to data subsets that they need to perform their work.

Least access

Only the [virtual network](#) in the SOC subscription has access to the SOC Storage account and key vault that archives the evidence. Authorized SOC team members can grant investigators temporary access to evidence in the SOC storage.

Evidence acquisition

Azure audit logs can document the evidence acquisition by recording the action of taking a VM disk snapshot. The logs include details such as who takes the snapshots and when they're taken.

Evidence integrity

Use [Automation](#) to move evidence to its final archive destination, without human intervention. This approach helps guarantee that evidence artifacts remain unaltered.

When you apply a legal hold policy to the destination storage, the evidence is immediately frozen as soon as it's written. A legal hold demonstrates that the chain of custody is fully maintained within Azure. It also indicates that there's no opportunity to tamper with the evidence from the time the disk images are on a live VM to when they are stored as evidence in the storage account.

Lastly, you can use the provided solution as an integrity mechanism to compute the hash values of the disk images. The supported hash algorithms are MD5, SHA256, SKEIN, and KECCAK (or SHA3).

Evidence production

Investigators need access to evidence so that they can perform analyses. This access must be tracked and explicitly authorized.

Provide investigators with a [shared access signatures \(SAS\) uniform resource identifier \(URI\)](#) storage key for accessing evidence. A SAS URI can generate relevant log information when it's created. You can obtain a copy of the evidence each time the SAS is used.

For example, if a legal team needs to transfer a preserved virtual hard drive, one of the two SOC team custodians generates a read-only SAS URI key that expires after eight hours. The SAS restricts access to the investigators within a specified time frame.

The SOC team must explicitly place the IP addresses of investigators that require access on an allowlist in the Storage firewall.

Finally, investigators need the BEKs archived in the SOC key vault to access the encrypted disk copies. An SOC team member must extract the BEKs and provide them via secure channels to the investigators.

Regional store

For compliance, some standards or regulations require evidence and the supporting infrastructure to be maintained in the same Azure region.

All the solution components, including the Storage account that archives evidence, are hosted in the same Azure region as the systems being investigated.

Operational Excellence

Operational Excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Design review checklist for Operational Excellence](#).

Monitoring and alerting

Azure provides services to all customers for monitoring and alerting about anomalies related to their subscriptions and resources. These services include:

- Microsoft Sentinel .
- Microsoft Defender for Cloud .
- Microsoft Defender for Storage.

 **Note**

The configuration of these services isn't described in this article.

Deploy this scenario

Follow the [chain of custody lab deployment](#) instructions to build and deploy this scenario in a laboratory environment.

The laboratory environment represents a simplified version of the architecture described in this article. You deploy two resource groups within the same subscription. The first resource group simulates the production environment, housing digital evidence, while the second resource group holds the SOC environment.

Select **Deploy to Azure** to deploy only the SOC resource group in a production environment.



Deploy to Azure



 **Note**

If you deploy the solution in a production environment, make sure that the system-assigned managed identity of the Automation account has the following permissions:

- A Contributor in the production resource group of the VM to be processed. This role creates the snapshots.
- A Key Vault Secrets User in the production key vault that holds the BEKs. This role reads the BEKs.

If the key vault has the firewall enabled, be sure that the public IP address of the hybrid runbook worker VM is allowed through the firewall.

Extended configuration

You can deploy a hybrid runbook worker on-premises or in different cloud environments.

In this scenario, you must customize the `Copy-VmDigitalEvidence` runbook to enable the capture of evidence in different target environments and archive them in storage.

 **Note**

The `Copy-VmDigitalEvidence` runbook provided in the [Deploy this scenario section](#) was developed and tested only in Azure. To extend the solution to other platforms, you must customize the runbook to work with those platforms.

Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal authors:

- [Fabio Masciotra](#) ↗ | Principal Consultant
- [Simone Savi](#) ↗ | Senior Consultant

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

For more information about Azure data-protection features, see:

- [Storage encryption for data at rest](#)
- [Overview of managed disk encryption options](#)
- [Store business-critical blob data with immutable storage in a WORM state](#)

For more information about Azure logging and auditing features, see:

- [Azure security logging and auditing](#)
- [Storage analytics logging](#)
- [Send Azure resource logs to Log Analytics workspaces, Event Hubs, or Storage](#)

For more information about Microsoft Azure compliance, see:

- [Azure compliance](#) ↗
- [Microsoft compliance offerings](#)

Related resource

- [Security architecture design](#)

Certificate lifecycle management on Azure

Azure Automation

Azure Event Grid

Azure Key Vault

In cybersecurity, setting up automatic certificate renewal is important to maintaining a secure and reliable environment. Failure to update or renew certificates in a timely manner exposes systems to vulnerabilities. Potentially vulnerable areas include:

- TLS/SSL certificates that are expired.
- Networks that are subject to potential breaches.
- Sensitive data that's unsecured.
- Services that go down for business-to-business processes.
- Brand reputation loss that compromises the integrity and confidentiality of digital transactions.

Azure Key Vault supports [automatic certificate renewal](#) issued by an integrated certification authority (CA) such as *DigiCert* or *GlobalSign*. For a nonintegrated CA, a [manual](#) approach is required.

This article bridges the gap by providing an automatic renewal process tailored to certificates from nonintegrated CAs. This process stores the new certificates in Key Vault, improves efficiency, enhances security, and simplifies deployment by integrating with various Azure resources.

An automatic renewal process reduces human error and minimizes service interruptions. When you automate certificate renewal, it accelerates the renewal process and decreases the likelihood of errors that might occur during manual handling. When you use the capabilities of Key Vault and its extensions, you can build an efficient automatic process to optimize operations and reliability.

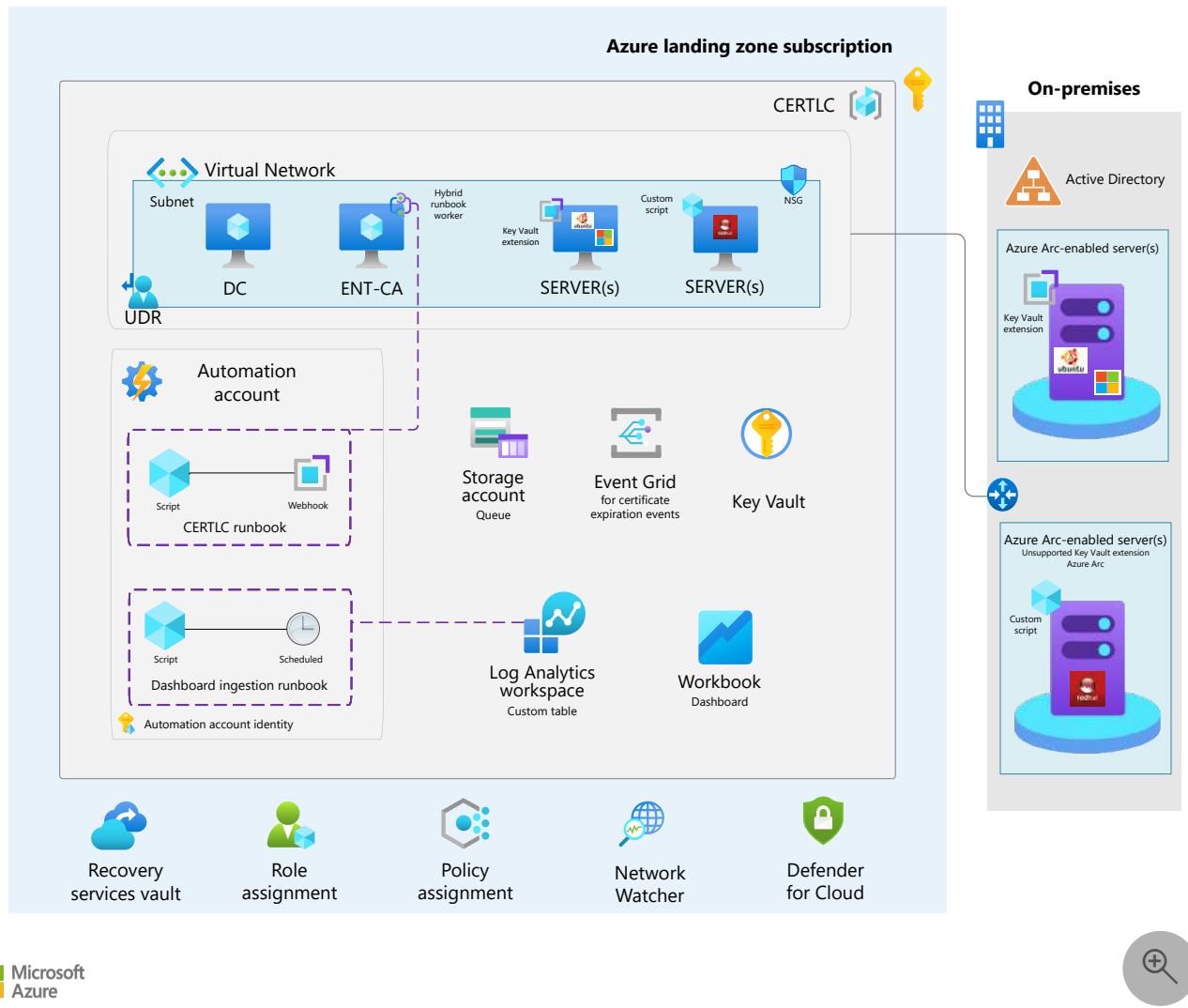
Automatic certificate renewal is the initial focus, but a broader objective is to enhance security across all areas of the process. This effort includes how to implement the principle of least privilege or similar access controls by using Key Vault. It also emphasizes the importance of robust logging and monitoring practices for Key Vault. This article highlights the importance of using Key Vault to fortify your entire certificate management lifecycle and demonstrates that the security benefits aren't limited to storing certificates.

You can use Key Vault and its automatic renewal process to continuously update certificates. Automatic renewal plays an important role in the deployment process and helps Azure services that integrate with Key Vault benefit from up-to-date certificates. This article provides insight

into how continuous renewal and accessibility contribute to the overall deployment efficiency and reliability of Azure services.

Architecture

The following diagram provides an overview of the underlying architecture that powers this solution.



Download a [Visio file](#) of this architecture.

The Azure environment comprises the following platform as a service (PaaS) resources:

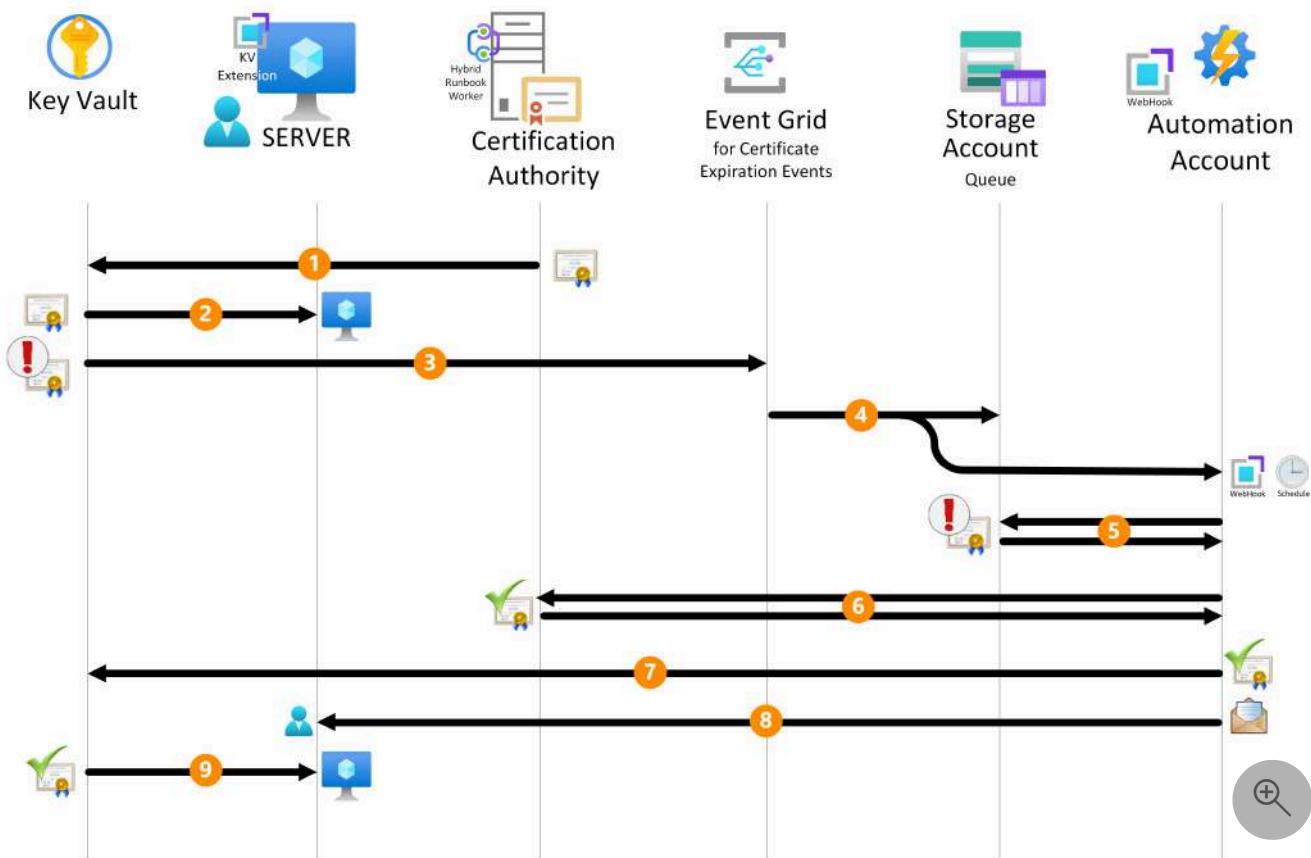
- A key vault that's dedicated to storing certificates issued only by the same nonintegrated CA
- An Azure Event Grid system topic
- A storage account queue
- An Azure Automation account that exposes a webhook targeted by Event Grid

To monitor the process and status of expired and expiring certificates, Log Analytics stores the data, and the workspace presents it in the form of tabular and graphical dashboards.

This scenario assumes that an existing public key infrastructure (PKI) is already in place and consists of a Microsoft Enterprise CA joined to a domain in Microsoft Entra ID. Both the PKI and the Active Directory domain can reside on Azure or on on-premises servers that are configured for certificate renewal.

The virtual machines (VMs) that have certificates to monitor renewal don't need to be joined to Active Directory or Microsoft Entra ID. The sole requirement is for the CA and the hybrid worker, if it's located on a different VM than the CA, to be joined to Active Directory.

The following diagram shows the automatic workflow for certificate renewal within the Azure ecosystem.



Workflow

The following workflow corresponds to the previous diagram:

- 1. Key Vault configuration:** The initial phase of the renewal process entails storing the certificate object in the designated Certificates section of the key vault.

Although not mandatory, you can set up custom email notifications by tagging the certificate with the recipient's email address. Tagging the certificate ensures timely

notifications when the renewal process completes. If multiple recipients are necessary, separate their email addresses by a comma or a semicolon. The tag name for this purpose is *Recipient*, and its value is one or more email addresses of the designated administrators.

When you use tags instead of [built-in certificate notifications](#), you can apply notifications to a specific certificate with a designated recipient. Built-in certificate notifications apply indiscriminately to all certificates within the key vault and use the same recipient for all.

You can integrate built-in notifications with the solution but use a different approach. Built-in notifications can only notify about an upcoming certificate expiration, but the tags can send notifications when the certificate renews on the internal CA and when it's available in Key Vault.

2. Key Vault extension configuration: You must equip the servers that need to use the certificates with the Key Vault extension, a versatile tool compatible with [Windows](#) and [Linux](#) systems. Azure infrastructure as a service (IaaS) servers and on-premises or other cloud servers that integrate through [Azure Arc](#) are supported. Configure the Key Vault extension to periodically poll Key Vault for any updated certificates. The polling interval is customizable and flexible so it can align with specific operational requirements.

 **Note**

The Key Vault extension isn't available on Linux RedHat and CentOS. To extend the solution to these systems, schedule the

[script_for_not_supported_ARC_on_Linux_distro](#) ↗ script that periodically checks Key Vault for certificate updates and applies them to the server. The script can run on Azure native VMs (IaaS) and on-premises servers integrated with Azure Arc.

3. Event Grid integration: As a certificate approaches expiration, two Event Grid subscriptions intercept this important lifetime event from the key vault.

4. Event Grid triggers: One Event Grid subscription sends certificate renewal information to a storage account queue. The other subscription triggers the launch of a runbook through the configured webhook in the Automation account. If the runbook fails to renew the certificate, or if the CA is unavailable, a scheduled process retries renewing the runbook from that point until the queue clears. This process makes the solution robust.

To enhance the solution's resiliency, set up a [dead-letter location](#) mechanism. It manages potential errors that might occur during the message's transit from Event Grid to the subscription targets, the storage queue, and the webhook.

5. **Storage account queue:** The runbook launches within the CA server that's configured as an Automation Hybrid Runbook Worker. It receives all messages in the storage account queue that contain the name of the expiring certificate and the key vault that hosts the runbook. The following steps occur for each message in the queue.

6. **Certificate renewal:** The script in the runbook connects to Azure to retrieve the certificate's template name that you set up during generation. The template is the configuration component of the certification authority that defines the attributes and purpose of the certificates to be generated.

After the script interfaces with Key Vault, it initiates a certificate renewal request. This request triggers Key Vault to generate a certificate signing request (CSR) and applies the same template that generated the original certificate. This process ensures that the renewed certificate aligns with the predefined security policies. For more information about security in the authentication and authorization process, see the [Security](#) section.

The script downloads the CSR and submits it to the CA.

The CA generates a new x509 certificate based on the correct template and sends it back to the script. This step ensures that the renewed certificate aligns with the predefined security policies.

7. **Certificate merging and Key Vault update:** The script merges the renewed certificate back into the key vault. This step finalizes the update process and removes the message from the queue. Throughout the entire process, the private key of the certificate is never extracted from the key vault.

8. **Monitoring and email notification:** All operations that various Azure components run, such as an Automation account, Key Vault, a storage account queue, and Event Grid, are logged within the Azure Monitor Logs workspace to enable monitoring. After the certificate merges into the key vault, the script sends an email message to administrators to notify them of the outcome.

9. **Certificate retrieval:** The Key Vault extension on the server plays an important role during this phase. It automatically downloads the latest version of the certificate from the key vault into the local store of the server that's using the certificate. You can configure multiple servers with the Key Vault extension to retrieve the same certificate (wildcard or with multiple Subject Alternative Name (SAN) certificates) from the key vault.

For Linux distributions where the Key Vault extension can't be installed, schedule the [script_for_not_supported_ARC_on_Linux_distro](#) script to achieve the same functionality as the extension.

Components

The solution uses various components to handle automatic certificate renewal on Azure. The following sections describe each component and its specific purpose.

Key Vault extension

The Key Vault extension is a tool installed on servers to automate certificate retrieval from Key Vault. This extension plays a vital role in automating certificate renewal and must be installed on servers that require the automation. In this architecture, this extension periodically polls Key Vault for updated certificates and automatically installs them on the server.

- For more information about installation procedures on Windows servers, see [Key Vault extension for Windows](#).
- For more information about installation steps for Linux servers, see [Key Vault extension for Linux](#).
- For more information about Azure Arc-enabled servers, see [Key Vault extension for Azure Arc-enabled servers](#).

! Note

You can run the following sample scripts from Azure Cloud Shell to configure the Key Vault extension:

- [Key Vault extension for Windows servers](#)
- [Key Vault extension for Linux servers](#)
- [Key Vault extension for Azure Arc-enabled Windows servers](#)
- [Key Vault extension for Azure Arc-enabled Linux servers](#)

The Key Vault extension configuration parameters include:

- **Key Vault name:** The key vault that contains the certificate for renewal.
- **Certificate name:** The name of the certificate to be renewed.
- **Certificate store, name, and location:** The certificate store where the certificate is stored. On Windows servers, the default value for *Name* is `My` and *Location* is `LocalMachine`, which is the personal certificate store of the computer. On Linux servers, you can specify a file system path, assuming that the default value is `AzureKeyVault`, which is the certificate store for Key Vault.

- **linkOnRenewal**: A flag that indicates whether the certificate should be linked to the server on renewal. If it's set to `true` on Windows machines, it copies the new certificate in the store and links it to the old certificate, which effectively rebinds the certificate. The default value is `false`, so an explicit binding is required.
- **pollingIntervalInS**: This value indicates the polling interval for the Key Vault extension to check for certificate updates. The default value is `3600` seconds (1 hour).
- **authenticationSetting**: The authentication setting for the Key Vault extension. For Azure servers, you can omit this setting, so the system-assigned managed identity of the VM is used against the key vault. For on-premises servers, specify the setting `msiEndpoint = "http://localhost:40342/metadata/identity"` so that the service principal that's associated with the computer object created during the Azure Arc onboarding is used.

 **Note**

Specify the Key Vault extension parameters only during the initial setup. This approach ensures that they don't undergo any changes throughout the renewal process.

Automation account

An Automation account is a cloud-based service that automates tasks via runbooks. In this architecture, it hosts the PowerShell runbook that renews certificates and is triggered by Event Grid via a webhook. You need to configure the account with a runbook by using the [PowerShell script](#).

You also need to create a Hybrid Worker Group. Associate the Hybrid Worker Group with a Windows Server member of the same Active Directory domain of the CA, ideally the CA itself, for launching runbooks.

The runbook must have an associated [webhook](#) initiated from the Hybrid Runbook Worker. Configure the webhook URL in the event subscription of the Event Grid system topic.

Storage account queue

The storage account queue is a message queue within Azure Storage. In this architecture, it stores the messages that contain the name of the certificate being renewed and the key vault that contains the certificate. Configure the storage account queue in the event subscription of the Event Grid system topic. The queue handles decoupling the script from the certificate expiration notification event. It supports persisting the event within a queue message. This

approach helps ensure that the renewal process for certificates is repeated through scheduled jobs even if problems occur during the script's run.

Hybrid Runbook Worker

The Hybrid Runbook Worker allows runbooks to run on-premises or non-Azure machines. In this architecture, it runs the certificate renewal script on a Windows Server in the same Windows Server Active Directory domain as the certificate authority (CA), ideally on the CA itself. It plays a vital role in using runbooks. You need to install the Hybrid Runbook Worker by using the [Azure Hybrid Worker extension](#) method, which supports new installation.

Key Vault

Key Vault is a secure repository for secrets, keys, and certificates. In this architecture, it stores certificates from a nonintegrated CA and emits expiration events that trigger the renewal workflow. The Event Grid system topic is integrated with the webhook of the Automation account.

Event Grid

Event Grid is an event-routing service. In this architecture, it monitors certificate expiration events and triggers actions such as launching the renewal runbook and posting messages to the storage queue. To configure Event Grid, set up the system topic and event subscription to monitor relevant events. Relevant events include certificate expiration alerts, actions triggered within the automation workflow, and messages posted in the storage account queue.

Configure the Event Grid system topic with the following parameters:

- **Source:** The name of the key vault that contains the certificates.
- **Source type:** The type of the source. For example, the source type for this solution is `Azure Key Vault`.
- **Event types:** The event type to be monitored. For example, the event type for this solution is `Microsoft.KeyVault.CertificateNearExpiry`. This event triggers when a certificate is near expiration.
- **Subscription for webhook:**
 - **Subscription name:** The name of the event subscription.
 - **Endpoint type:** The type of endpoint to be used. For example, the endpoint type for this solution is `Webhook`.

- **Endpoint:** The URL of the webhook that's associated with the Automation account runbook. For more information, see the [Automation account](#) section.
- **Subscription for StorageQueue:**
 - **Subscription name:** The name of the event subscription.
 - **Endpoint type:** The type of endpoint to be used. For example, the endpoint type for this solution is `StorageQueue`.
 - **Endpoint:** The storage account queue.

Log Analytics workspace and Azure workbook

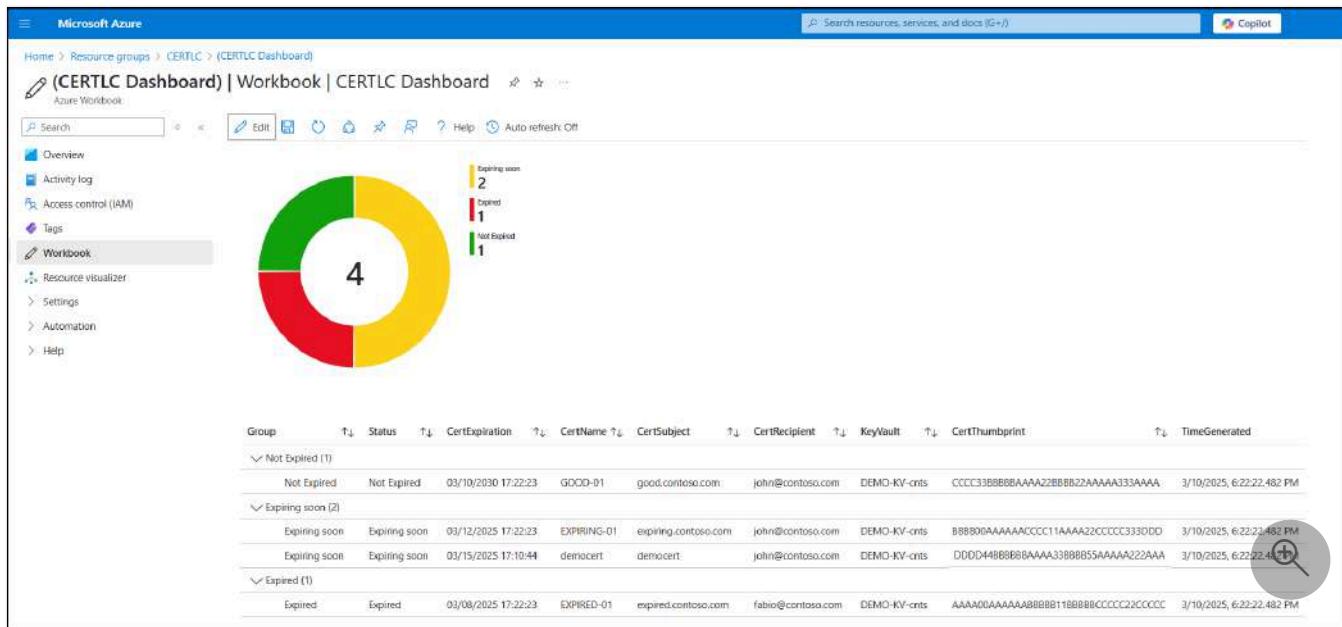
These tools collect and store logs from Azure services. This solution uses Log Analytics workspace and Azure workbook to enhance monitoring and visualization of certificate statuses stored in Key Vault. These components play a crucial role in maintaining visibility into certificate health:

- **Log Analytics workspace** collects and stores data about certificate states. It identifies whether certificates are expired, expiring soon, or still valid.
- **Azure workbook** retrieves data from the Log Analytics workspace and presents it in a dashboard with visual representations, like pie charts and detailed tables. It categorizes certificates into *Not Expired* (green), *Expiring Soon* (yellow), and *Expired* (red).

The following components retrieve and present certificate information in the workbook:

- A **data ingestion runbook** runs directly from Azure and doesn't require the context of a Hybrid Worker. It retrieves certificate data from Key Vault and sends it to a custom table that's defined in the Log Analytics workspace. The runbook runs on a scheduled cadence.
- A **workbook** queries the data from the custom table and displays it in both a pie chart and a detailed table. It highlights certificates based on their expiration status.

By integrating these components, your solution builds a more comprehensive approach to certificate lifecycle management.



Alternatives

This solution uses an Automation account to orchestrate the certificate renewal process and uses Hybrid Runbook Worker to provide the flexibility to integrate with a CA on-premises or in other clouds.

An alternative approach is to use Azure Logic Apps. The main difference between the two approaches is that the Automation account is a PaaS solution, and Logic Apps is a software as a service (SaaS) solution.

The main advantage of Logic Apps is that it's a fully managed service. You don't need to worry about the underlying infrastructure. Also, Logic Apps can easily integrate with external connectors. This capability expands the range of notification possibilities, such as engagement with Microsoft Teams or Microsoft 365.

Logic Apps doesn't have a feature that's similar to Hybrid Runbook Worker, which results in less flexible integration with the CA, so an Automation account is the preferred approach.

Scenario details

Every organization requires secure and efficient management of their certificate lifecycle. Failing to update a certificate before expiration can lead to service interruptions and incur significant costs for the business.

Enterprises typically operate complex IT infrastructures that involve multiple teams who are responsible for the certificate lifecycle. The manual nature of the certificate renewal process often introduces errors and consumes valuable time.

This solution addresses those challenges by automating certificate renewal issued by Microsoft Certificate Service. The service is widely used for various server applications such as web servers, SQL servers, and for encryption, nonrepudiation, signing purposes, and ensuring timely updates and secure certificate storage within Key Vault. The service's compatibility with Azure servers and on-premises servers supports flexible deployment.

Potential use cases

This solution caters to organizations across various industries that:

- Use Microsoft Certificate Service for server certificate generation.
- Require automation in the certificate renewal process to accelerate operations and minimize errors, which helps avoid business loss and service-level agreement (SLA) violations.
- Require secure certificate storage in repositories like Key Vault.

This architecture serves as a foundational deployment approach across application landing zone subscriptions.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

Security

Security provides assurances against deliberate attacks and the misuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

Within the Key Vault system, certificates are more securely stored as encrypted secrets and protected by Azure role-based access control (Azure RBAC).

Throughout the certificate renewal process, components that use identities are:

- The system account of the Hybrid Runbook Worker, which operates under the VM's account.
- The Key Vault extension, which uses the managed identity that's associated with the VM.
- The Automation account, which uses its designated managed identity.

The principle of least privilege is rigorously enforced across all identities engaged in the certificate renewal procedure.

The system account of the Hybrid Runbook Worker server must have the right to enroll certificates on one or more certificate templates that generate new certificates.

On the key vault that contains the certificates, the Automation account identity must have the `Key Vault Certificate Officer` role. Additionally, servers that require certificate access must have `Get` and `List` permissions within the Key Vault certificate store.

On the storage account queue, the Automation account identity must have the `Storage Queue Data Contributor`, `Reader and Data Access`, and `Reader` roles.

In scenarios where the Key Vault extension deploys on an Azure VM, the authentication occurs via the managed identity of the VM. However, when it's deployed on an Azure Arc-enabled server, authentication is handled by using a service principal. Both the managed identity and service principal must be assigned the Key Vault secret user role within the key vault that stores the certificate. You must use a secret role because the certificate is stored in the key vault as a secret.

Cost Optimization

Cost Optimization focuses on ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

This solution uses Azure PaaS solutions that operate under a pay-as-you-go framework to optimize cost. Expenses depend on the number of certificates that need renewal and the number of servers equipped with the Key Vault extension, which results in low overhead.

Expenses that result from the Key Vault extension and the Hybrid Runbook Worker depend on your installation choices and polling intervals. The cost of Event Grid corresponds to the volume of events generated by Key Vault. At the same time, the cost of the Automation account correlates with the number of runbooks that you use.

The cost of Key Vault depends on various factors, including your chosen SKU (Standard or Premium), the quantity of stored certificates, and the frequency of operations conducted on the certificates.

Similar considerations to the configurations described for Key Vault apply equally to the storage account. In this scenario, a Standard SKU with locally redundant storage replication suffices for the storage account. Generally, the cost of the storage account queue is minimal.

To estimate the cost of implementing this solution, use the [Azure pricing calculator](#). Input the services described in this article.

Operational Excellence

Operational Excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Design review checklist for Operational Excellence](#).

The automatic certificate renewal procedure securely stores certificates by way of standardized processes applicable across all certificates within the key vault.

Integrating with Event Grid triggers supplementary actions, such as notifying Microsoft Teams or Microsoft 365 and streamlining the renewal process. This integration significantly reduces certificate renewal time and mitigates the potential for errors that can lead to business disruptions and violations of SLAs.

Also, seamless integration with Azure Monitor, Microsoft Sentinel, Microsoft Copilot for Security, and Microsoft Defender for Cloud facilitates continuous monitoring of the certificate renewal process. It supports anomaly detection and ensures that robust security measures are maintained.

Deploy this scenario

Select the following button to deploy the environment described in this article. The deployment takes about two minutes to complete and creates a key vault, an Event Grid system topic configured with the two subscriptions, a storage account that contains the *certlc* queue, and an Automation account that contains the *runbook* and the *webhook* linked to Event Grid.



Deploy to Azure



You can find detailed information about the parameters needed for the deployment in the [code sample](#) portal.

Important

You can deploy a full lab environment to demonstrate the entire automatic certificate renewal workflow. Use the [code sample](#) to deploy the following resources:

- **Active Directory Domain Services (AD DS)** within a domain controller VM.
- **Active Directory Certificate Services (AD CS)** within a CA VM, joined to the domain, configured with a template, *WebServerShort*, for enrolling the certificates to renew.

- A Windows Simple Mail Transfer Protocol (SMTP) server installed on the same VM of the CA for sending email notifications. MailViewer also installs to verify the email notifications sent.
- The **Key Vault extension** installed on the VM of the domain controller for retrieving the renewed certificates from the Key Vault extension.



Deploy to Azure



Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal authors:

- [Fabio Masciotra](#) | Principal Consultant
- [Angelo Mazzucchi](#) | Principal Consultant

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

- [About Azure Key Vault](#)
- [Key Vault VM extension for Windows](#)
- [Key Vault VM extension for Linux](#)
- [What is Azure Automation?](#)
- [Azure Automation Hybrid Runbook Worker overview](#)
- [What is Event Grid?](#)

Multilayered protection for Azure virtual machine access

Microsoft Entra ID

Azure Bastion

Azure Role-based access control

Microsoft Defender for Cloud

Azure Key Vault

Solution ideas

This article describes a solution idea. Your cloud architect can use this guidance to help visualize the major components for a typical implementation of this architecture. Use this article as a starting point to design a well-architected solution that aligns with your workload's specific requirements.

This solution offers a multilayered strategy for protecting virtual machines (VMs) in Azure, ensuring accessibility while minimizing the attack surface for management and administrative purposes.

Aligned with Microsoft's security recommendation, this solution incorporates several protection mechanisms offered by Microsoft Azure and Entra services, adhering to the principles of secure by design, secure by default, and secure operations.

- **Secure by design.** The solution achieves non-persistent granular access to virtual machines by implementing the principle of least privilege and the concept of separation of duties. This ensures that authorization to the virtual machines is granted only for legitimate reasons, reducing the risk of unauthorized access.
- **Secure by default.** Inbound traffic to virtual machines is locked down, allowing connectivity only when needed. This default security posture minimizes exposure to many popular cyber-attacks such as brute-force and distributed denial-of-service (DDoS) attacks.
- **Secure operations.** It's critical to implement continuous monitoring and invest in improving of security controls to meet current and future threats. Use various Azure services and features such as Microsoft Entra Privileged Identity Management (PIM), the just-in-time (JIT) VM access feature of Microsoft Defender for Cloud, Azure Bastion, Azure role-based access control (Azure RBAC) custom roles. Optionally you should consider Microsoft Entra Conditional Access to regulate access to Azure resources and Azure Key Vault for storing virtual machine local passwords if not integrated with Entra ID or Active Directory Domain Services.

Potential use cases

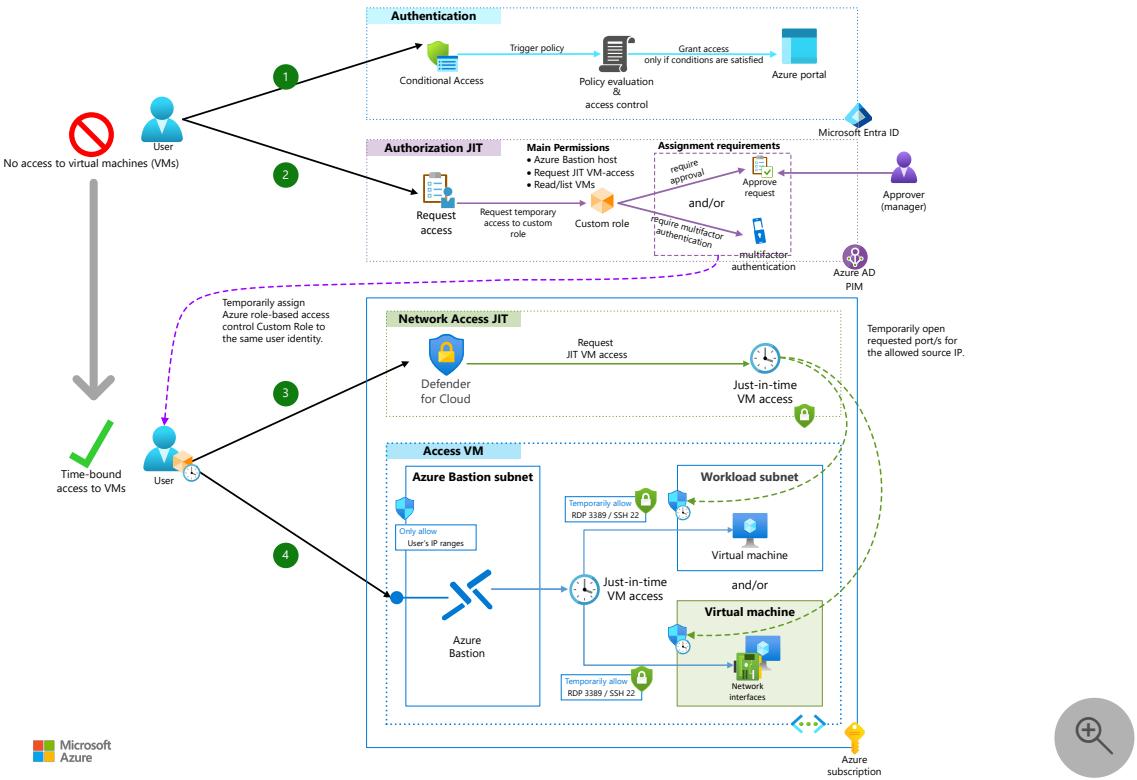
Defense in depth is the premise behind this architecture. This strategy challenges users with several lines of defense before granting the users access to VMs. The goal is to ensure that:

- Each user is verified.
- Each user has legitimate intentions.
- Communication is secure.
- Access to VMs in Azure is only provided when needed.

The defense in depth strategy and the solution in this article apply to many scenarios:

- An administrator needs to access an Azure VM under these circumstances:
 - The administrator needs to troubleshoot an issue, investigate behavior, or apply a critical update.
 - The administrator uses Remote Desktop Protocol (RDP) to access a Windows VM or secure shell (SSH) to access a Linux VM.
 - The access should include the minimum number of permissions required for performing the task.
 - The access should be valid for only a limited time.
 - After the access expires, the system should lock down the VM access to prevent malicious access attempts.
- Employees need access to a remote workstation that's hosted in Azure as a VM. The following conditions apply:
 - The employees should access the VM only during work hours.
 - The security system should consider requests to access the VM outside work hours unnecessary and malicious.
- Users would like to connect to Azure VM workloads. The system should approve connections that are only from managed and compliant devices.
- A system has experienced a tremendous number of brute-force attacks:
 - These attacks have targeted Azure VMs on RDP and SSH ports 3389 and 22.
 - The attacks have tried to guess the credentials.
 - The solution should prevent access ports such as 3389 and 22 from being exposed to the internet or on-premises environments.

Architecture



Download a [Visio file](#) of this architecture.

Dataflow

- 1. Authentication and access decisions:** The user is authenticated against Microsoft Entra ID to access the Azure portal, Azure REST APIs, Azure PowerShell, or the Azure CLI. If authentication succeeds, a Microsoft Entra Conditional Access policy takes effect. That policy verifies whether the user meets certain criteria. Examples include using a managed device or signing in from a known location. If the user fulfills the criteria, Conditional Access grants the user access to Azure through the Azure portal or another interface.
- 2. Identity-based just-in-time access:** During authorization, Microsoft Entra PIM assigns the user a custom role of type *eligible*. The eligibility is limited to required resources and is a *time-bound* role, not a *permanent* one. Within a specified time frame, the user requests activation of this role through the Azure PIM interface. That request can trigger other actions, such as starting an approval workflow or prompting the user for multifactor authentication to verify identity. In an approval workflow, another person needs to approve the request. Otherwise the user isn't assigned the custom role and can't continue to the next step.
- 3. Network based just-in-time access:** After authentication and authorization, the custom role is temporarily linked to the user's identity. The user then requests JIT VM access. That access opens a connection from the Azure Bastion subnet on port 3389 for RDP or port

22 for SSH. The connection runs directly to the VM network interface card (NIC) or the VM NIC subnet. Azure Bastion opens an internal RDP session by using that connection. The session is limited to the Azure virtual network and isn't exposed to the public internet.

4. Connecting to the Azure VM: The user accesses Azure Bastion by using a temporary token. Through this service, the user establishes an indirect RDP connection to the Azure VM. The connection only works for a limited amount of time. The user might retrieve the password from an Azure Key Vault, if the password was stored as a secret in the Key Vault, and sufficient RBAC permissions are configured to limit access to the appropriate user account.

Components

This solution uses the following components:

- [Azure Virtual Machines](#) is an infrastructure as a service (IaaS) offering that provides scalable compute resources. In this architecture, Azure VMs host production workloads while minimizing exposure to threats through layered security controls.
- [Microsoft Entra ID](#) is a cloud-based identity service that manages access to Azure and other cloud applications. In this architecture, it authenticates users and enforces access policies to ensure secure entry into Azure resources.
- [Microsoft Entra PIM](#) is a service that controls and monitors privileged access to resources. In this architecture, PIM limits permanent admin access to standard and custom privileged roles and enables just-in-time (JIT) identity-based access to custom roles.
- [JIT VM access](#) is a Defender for Cloud feature that restricts network access to VMs. In this architecture, JIT minimizes the attack surface by applying deny rules and only allowing temporary access when requested. When a user requests access to the VM, the service adds a temporary allow rule to the network security group. Because the allow rule has higher priority than the deny rule, the user can connect to the VM. Azure Bastion works best for connecting to the VM. But the user can also use a direct RDP or SSH session.
- [Azure RBAC](#) is an authorization system for managing access to Azure resources. In this architecture, [Azure RBAC custom roles](#) enforce the principle of least privilege by granting only necessary permissions for VM access. You can use them to assign permissions at levels that meet your organization's needs. To access a VM in this solution, the user gets permissions for the following actions:
 - Using Azure Bastion
 - Requesting JIT VM access in Defender for Cloud
 - Reading or listing VMs

- [Microsoft Entra Conditional Access](#) is a policy-based access control tool. In this architecture, Conditional Access ensures that only authenticated users from trusted devices or locations can access Azure resources. Conditional Access policies support the [Zero Trust](#) security model.
- [Azure Bastion](#) is a managed service that provides RDP and SSH connectivity to VMs over HTTPS. In this architecture, Azure Bastion connects users who use Microsoft Edge or another internet browser for HTTPS, or secured traffic on port 443. Azure Bastion sets up the RDP connection to the VM. RDP and SSH ports aren't exposed to the internet or the user's origin.

Azure Bastion is optional in this solution. Users can connect directly to Azure VMs by using the RDP protocol. If you do configure Azure Bastion in an Azure virtual network, set up a separate subnet called `AzureBastionSubnet`. Then associate a network security group with that subnet. In that group, specify a source for HTTPS traffic such as the user's on-premises IP classless inter-domain routing (CIDR) block. This configuration blocks connections that don't come from the user's on-premises environment.

- [Key Vault](#) is a service for storing secrets, keys, and certificates. In this architecture, Key Vault stores VM passwords as secrets and integrates with Azure Bastion to allow retrieval by authorized users. You can configure the secret RBAC so that only the user account that accesses the VM can retrieve it. Retrieving the password value from the key vault can be done through Azure APIs (such as using the Azure CLI) or from the Azure portal.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Husam Hilal](#) | Senior Cloud Solution Architect

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

- [Activate my Azure resource roles in Privileged Identity Management](#)
- [Understanding just-in-time \(JIT\) VM access](#)
- [Configure Bastion and connect to a Windows VM through a browser](#)
- [\[Secure user sign-in events with Microsoft Entra multifactor authentication\]\[Secure user sign-in events with Azure AD Multi-Factor Authentication\]](#)

Related resource

- [Azure Virtual Machines baseline architecture](#)

Map threats to your IT environment

Azure

Office 365

Solution ideas

This article describes a solution idea. Your cloud architect can use this guidance to help visualize the major components for a typical implementation of this architecture. Use this article as a starting point to design a well-architected solution that aligns with your workload's specific requirements.

This article outlines how to diagram your organization's core IT environment and create a threat map. These diagrams are valuable tools for planning and building a robust defensive security layer. Understanding your IT environment and its architecture is crucial for identifying the security services needed to provide adequate protection.

Computer systems hold information that is not only valuable to the organizations that generate it but also to malicious actors. These actors, whether individuals or groups, engage in harmful activities aimed at compromising or damaging the computers, devices, systems, and networks of companies. Their goal is often to steal or corrupt sensitive data using threats like malware or brute force attacks.

In this article, we explore a method for mapping threats to your IT environment, enabling you to plan the implementation of Microsoft security services as part of your security strategy.

The good news is that you don't need to create a threat map from scratch. The MITRE ATT&CK matrix offers an excellent resource to help you develop one. MITRE ATT&CK is a global knowledge base that maps real-world threats based on observed tactics and techniques. The MITRE Corporation documents every known threat in detail, providing valuable insights into how these threats operate and how you can defend against them. This publicly accessible resource is available online at [MITRE ATT&CK®](#).

In this article, we use a subset of these threats to illustrate how you can map threats to your IT environment.

Potential use cases

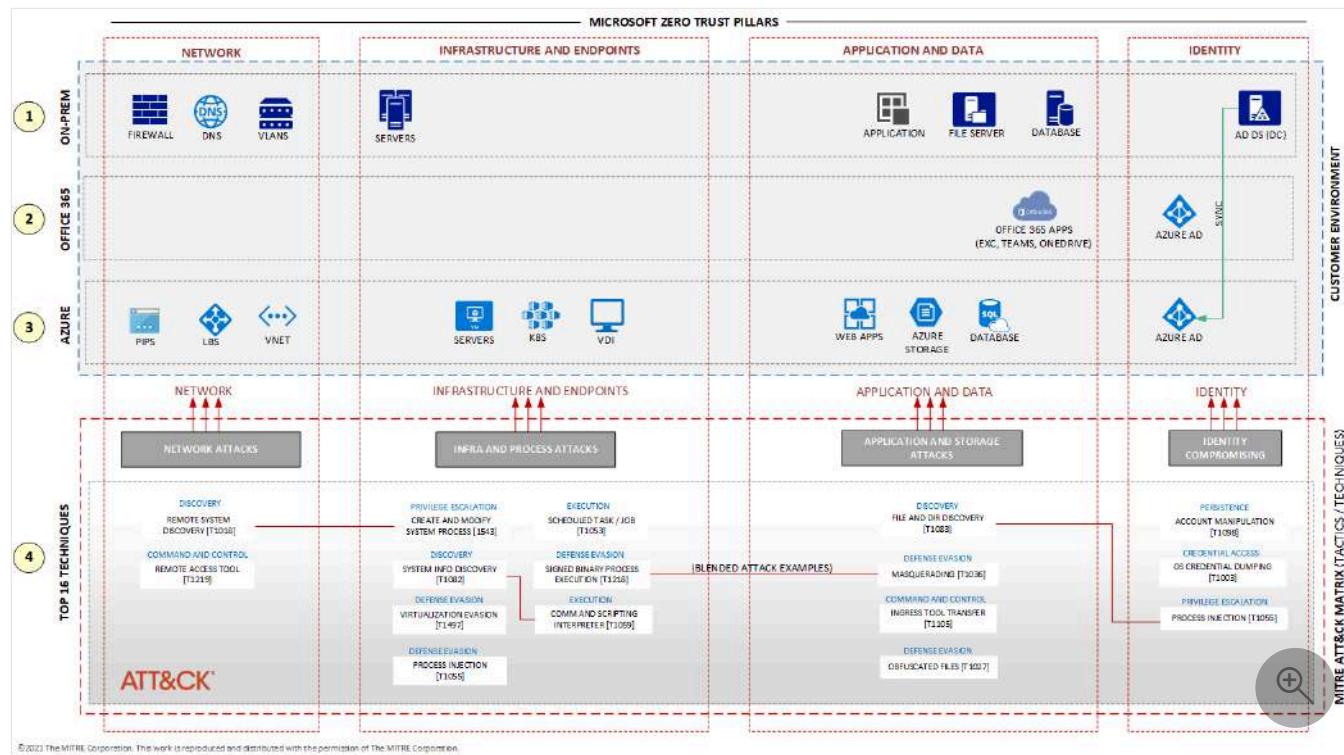
Some threats are common across all industries, such as ransomware, DDoS attacks, cross-site scripting, and SQL injection. However, many organizations face specific threats unique to their industry or based on past cyberattacks they've encountered. The diagram in this article can help you map those threats for your organization by identifying the areas most likely to be

targeted by malicious actors. Creating a threat map enables you to plan the necessary defense layers for a more secure environment.

You can adapt this diagram to model different combinations of attacks and better understand how to prevent and mitigate them. While the MITRE ATT&CK framework is a useful reference, it's not required. Microsoft Sentinel and other Microsoft security services also collaborate with MITRE to provide valuable insights into various threats.

Some organizations use Cyber Kill Chain®, a methodology from Lockheed Martin, to map and understand how an attack or a series of attacks are performed against an IT environment. Cyber Kill Chain organizes threats and attacks by considering fewer tactics and techniques than the MITRE ATT&CK framework. Still, it's effective in helping you to understand threats and how they might be executed. For more information about this methodology, see [Cyber Kill Chain](#).

Architecture



Download a [Visio file](#) of this architecture.

©2021 The MITRE Corporation. This work is reproduced and distributed with the permission of The MITRE Corporation.

For the IT environment of organizations, we specify the components only for Azure and Microsoft 365. Your specific IT environment might include devices, appliances, and technologies from different technology providers.

For the Azure environment, the diagram shows the components that are listed in the following table.

[Expand table](#)

Label	Documentation
VNET	What is Azure Virtual Network?
LBS	What is Azure Load Balancer?
PIPS	Public IP addresses
SERVERS	Virtual Machines
K8S	Azure Kubernetes Service
VDI	What is Azure Virtual Desktop?
WEB APPS	App Service overview
AZURE STORAGE	Introduction to Azure Storage
DB	What is Azure SQL Database?
Microsoft Entra ID	What is Microsoft Entra ID?

The diagram represents Microsoft 365 through the components listed in the following table.

[Expand table](#)

Label	Description	Documentation
OFFICE 365	Microsoft 365 services (formerly Office 365). The applications that Microsoft 365 makes available depends on the type of license.	Microsoft 365 - Subscription for Office Apps

Label	Description	Documentation
Microsoft Entra ID	Microsoft Entra ID, the same one utilized by Azure. Many companies use the same Microsoft Entra service for Azure and Microsoft 365.	What is Microsoft Entra ID?

Workflow

To help you understand which part of your IT environment those threats are likely to attack, the architecture diagram in this article is based on a typical IT environment for an organization that has on-premises systems, a Microsoft 365 subscription, and an Azure subscription. The resources in each of these layers are services that are common to many companies. They're classified in the diagram according to the pillars of Microsoft Zero Trust: network, infrastructure, endpoint, application, data, and identity. For more information about Zero Trust, see [Embrace proactive security with Zero Trust](#).

The architecture diagram includes the following layers:

1. On-premises

The diagram includes some essential services such as servers (VMs), network appliances, and Domain Name System (DNS). It includes common applications that are found in most IT environments and run on virtual machines (VMs) or physical servers. It also includes various types of databases, both SQL and non-SQL. Organizations usually have a file server that shares files throughout the company. Lastly, the Active Directory Domain Service, a widespread infrastructure component, handles user credentials. The diagram includes all these components in the on-premises environment.

2. Office 365 environment

This example environment contains traditional office applications, such as Word, Excel, PowerPoint, Outlook, and OneNote. Depending on the type of license, it might also include other applications, such as OneDrive, Exchange, Sharepoint, and Teams. In the diagram, these are represented by an icon for Microsoft 365 (formerly Office 365) apps and an icon for Microsoft Entra ID. Users must be authenticated to obtain access to Microsoft 365 applications, and Microsoft Entra ID acts as the identity provider. Microsoft 365 authenticates users against the same type of Microsoft Entra ID that Azure uses. In most organizations, the [Microsoft Entra ID tenant](#) is the same for both Azure and Microsoft 365.

3. Azure environment

This layer represents Azure public cloud services, including VMs, virtual networks, platforms as services, web applications, databases, storage, identity services, and more. For more information about Azure, see [Azure documentation](#).

4. MITRE ATT&CK tactics and techniques

This diagram shows the top 16 threats, according to the tactics and techniques as published by The MITRE Corporation. In red lines, you can see an example of a blended attack, which means that a malicious actor might coordinate multiple attacks simultaneously.

How to use the MITRE ATT&CK framework

You can start with a simple search for the name of the threat or of the attack code on the main web page, [MITRE ATT&CK®](#).

You can also browse threats on the tactics or techniques pages:

- [Enterprise tactics](#)
- [Enterprise techniques](#)

You can still use [MITRE ATT&CK® Navigator](#), an intuitive tool provided by MITRE that helps you discover tactics, techniques, and details about threats.

Components

The example architecture in this article uses the following Azure components:

- [Microsoft Entra ID](#) is a cloud-based identity and access management service that enables secure access to internal and external resources. In this architecture, it authenticates users for both Azure and Microsoft 365 services. It serves as the central identity provider across the environment.
- [Azure Virtual Network](#) is a networking service in Azure that enables secure communication between Azure resources, the internet, and on-premises networks. In this architecture, it provides isolated and scalable network infrastructure for hosting workloads and enforcing traffic control.
- [Azure Load Balancer](#) is a high-performance layer-4 load balancing service for Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) traffic. In this architecture, it ensures high availability and scalability by distributing inbound and outbound traffic across VMs and services.

- [Azure Virtual Machines](#) is an infrastructure as a service (IaaS) offering that provides flexible, on-demand compute resources. In this architecture, VMs host applications and services that are part of the organization's IT environment and are subject to threat mapping.
- [Azure Kubernetes service \(AKS\)](#) is a managed Kubernetes service for deploying and managing containerized applications. In this architecture, it runs containerized applications and supports enterprise-grade security and governance as part of the threat surface.
- [Virtual Desktop](#) is a desktop and app virtualization service that runs on the cloud to provide desktops for remote users. In this architecture, it provides secure access for remote users and is included in the threat map as a potential attack vector.
- The [Web Apps feature of Azure App Service](#) hosts web applications, REST APIs, and mobile back ends. You can develop in your chosen language. Applications run and scale with ease on both Windows and Linux-based environments. In this architecture, Web Apps hosts HTTP-based applications that are protected via integrated security features such as Transport Layer Security (TLS) and private endpoints.
- [Azure Storage](#) is a scalable and secure storage service for various data objects in the cloud, including object, blob, file, disk, queue, and table storage. Azure Storage encrypts all data written to a Storage account. It provides fine-grained control over access to your data. In this architecture, it stores application and system data and is included in the threat map because of its role in data protection and access control.
- [SQL Database](#) is a managed relational database engine that automates patching, backups, and monitoring. In this architecture, it stores structured data and supports built-in security and compliance features to mitigate threats.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Rudnei Oliveira](#) | Senior Azure Security Engineer

Other contributors:

- [Gary Moore](#) | Programmer/Writer
- [Andrew Nathan](#) | Senior Customer Engineering Manager

Next steps

This document refers to some services, technologies, and terminologies. You can find more information about them in the following resources:

- [MITRE ATT&CK® ↗](#)
- [ATT&CK® Navigator\) ↗](#)
- [Public Preview: The MITRE ATT&CK Framework Blade in Microsoft Sentinel ↗](#), a post from the Azure Cloud & AI Domain Blog
- [The Cyber Kill Chain® ↗](#)
- [Embrace proactive security with Zero Trust ↗](#)
- [Blended threat ↗](#) on Wikipedia
- [How cyberattacks are changing according to new Microsoft Digital Defense Report ↗](#) from Microsoft Security Blog

Related resources

For more information about this reference architecture, see the other articles in this series:

- Part 2: [Build the first layer of defense with Azure Security services](#)
- Part 3: [Build the second layer of defense with Microsoft Defender XDR Security services](#)
- Part 4: [Integrate Azure and Microsoft Defender XDR security services](#)

Build the first layer of defense with Azure Security services

Azure

Microsoft Entra ID

Solution ideas

This article describes a solution idea. Your cloud architect can use this guidance to help visualize the major components for a typical implementation of this architecture. Use this article as a starting point to design a well-architected solution that aligns with your workload's specific requirements.

You can use a various Azure services to create a complete IT infrastructure for your organization. Azure also provides security services that can help you protect your infrastructure. By using Azure security solutions, you can enhance your IT environment's security posture, mitigate vulnerabilities, and protect against breaches via a well-architected solution that's based on Microsoft best practices.

Although some security services incur associated costs, many are available at no additional charge. Free services include network security groups (NSGs), storage encryption, TLS/SSL, shared access signature tokens, and more. This article focuses on these cost-free services.

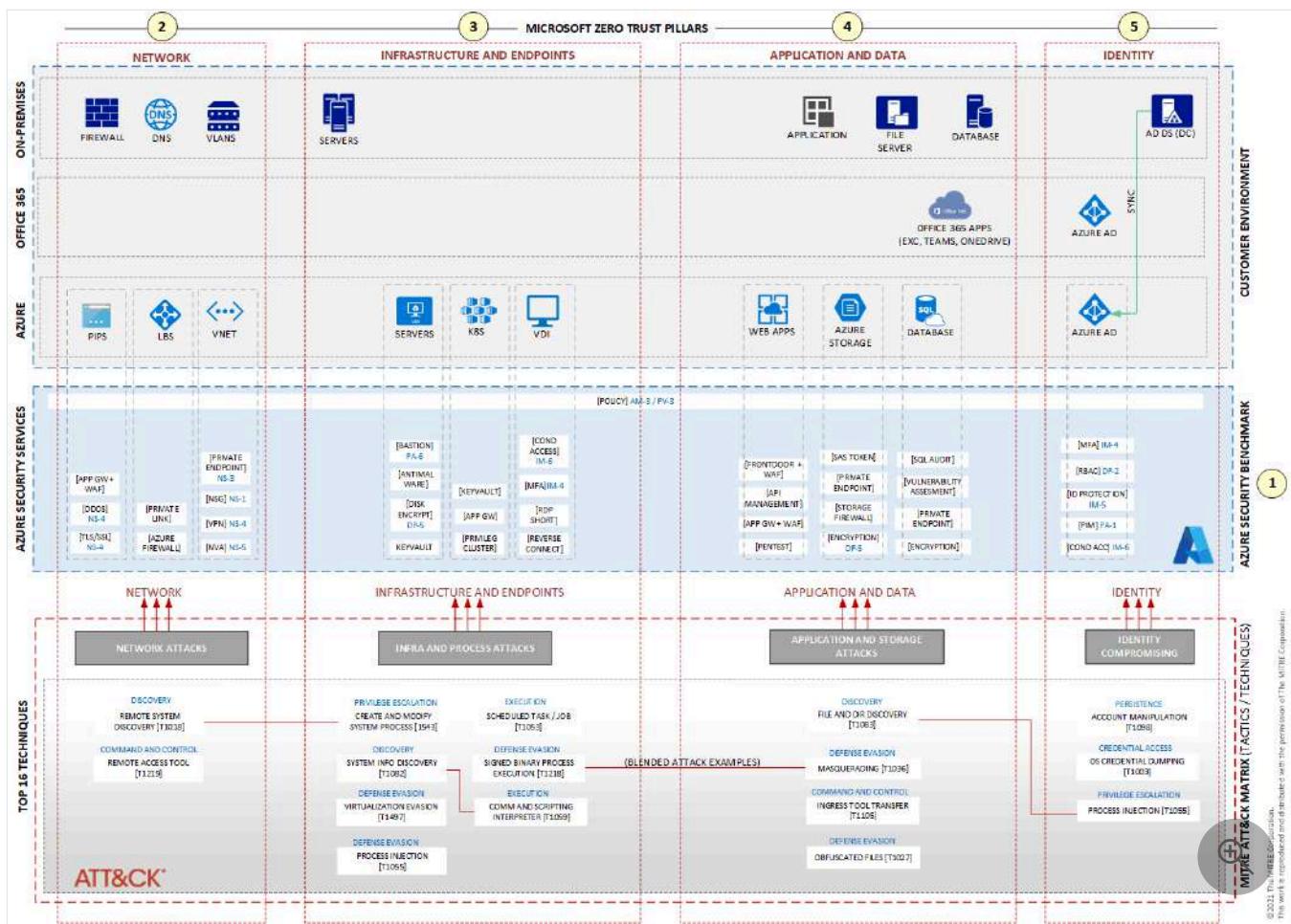
This article is the third in a series of five. To review the previous two articles in this series, including the introduction and a review of how you can map threats against an IT environment, see the following article:

- [Map threats to your IT environment](#)

Potential use cases

This article organizes Azure security services by Azure resource so you can focus on specific threats that target resources like virtual machines (VMs), operating systems, Azure networks, or applications, in addition to attacks that can compromise users and passwords. The following diagram can help you identify the Azure security services that help protect resources and user identities against these types of threats.

Architecture



Download a [Visio file](#) of this architecture.

©2021 The MITRE Corporation. This work is reproduced and distributed with the permission of The MITRE Corporation.

The Azure security layer in this diagram is based on Azure Security Benchmark (ASB) v3, which is a set of security rules that are implemented through Azure policies. ASB is based on a combination of rules from [CIS Center for Internet Security](#) and [National Institute of Standards and Technology](#). For more information about ASB, see [Overview of the Azure Security Benchmark v3](#).

The diagram doesn't include every Azure security service available, but it does highlight the services that are used most commonly. All the security services shown in the architectural diagram can be combined and configured to work together with your IT environment and your organization's specific security needs.

Workflow

This section describes the components and services that appear in the diagram. Many of those are labeled with their ASB control codes, in addition to their abbreviated labels. The control codes correspond to the control domains that are listed in [Controls](#).

1. Azure Security Benchmark

Each security control refers to one or more specific Azure security services. The architecture reference in this article shows some of them and their control numbers according to the ASB documentation. The controls include:

- Network security
- Identity management
- Privileged access
- Data protection
- Asset management
- Logging and threat detection
- Incident response
- Posture and vulnerability management
- Endpoint security
- Backup and recovery
- DevOps security
- Governance and strategy

For more information about security controls, see [Overview of the Azure Security Benchmark \(v3\)](#).

2. Network

The following table describes the network services in the diagram.

[Expand table](#)

Label	Description	Documentation
NSG	A free service that you attach to a network interface or subnet. An NSG allows you to filter TCP or UDP protocol traffic by using IP address ranges and ports for inbound and outbound connections.	Network security groups
VPN	A virtual private network (VPN) gateway that delivers a tunnel with IPSEC (IKE v1/v2) protection.	VPN Gateway
Azure Firewall	A platform as a service (PaaS) that delivers protection in layer 4 and is attached to an	What is Azure Firewall?

Label	Description	Documentation
	entire virtual network.	
App GW + WAF	Azure Application Gateway with Web Application Firewall (WAF). Application Gateway is a load balancer for web traffic that works in layer 7 and adds WAF to protect applications that use HTTP and HTTPS.	What is Azure Application Gateway?
NVA	Network virtual appliance (NVA). A virtual security service from the marketplace that's provisioned on VMs on Azure.	Network virtual appliances
DDOS	DDoS protection implemented on the virtual network to help you mitigate different types of DDoS attacks.	Azure DDoS Network Protection overview
TLS/SSL	TLS/SSL deliver encryption in transit for most Azure services that exchange information, such as Azure Storage and Web Apps.	Configure end-to-end TLS by using Application Gateway with PowerShell
Private Link	Service that allows you to create a private network for an Azure service that initially is exposed to the internet.	What is Azure Private Link?
Private endpoint	Creates a network interface and attaches it to the Azure service. Private Endpoint is part of Private Link. This configuration lets the service, by using a private endpoint, be part of your virtual network.	What is a private endpoint?

3. Infrastructure and endpoints

The following table describes infrastructure and endpoint services that are shown in the diagram.

[] [Expand table](#)

Label	Description	Documentation
Bastion	Bastion provides jump server functionality. This service allows you to access your VMs through remote desktop protocol (RDP) or SSH without exposing your VMs to the internet.	What is Azure Bastion?
Antimalware	Microsoft Defender provides antimalware service and is part of Windows 10, Windows 11, Windows Server 2016, and Windows Server 2019.	Microsoft Defender Antivirus in Windows
Disk encrypt	Disk Encryption allows you to encrypt the disk of a VM.	Azure Disk Encryption for Windows VMs
Keyvault	Key Vault, a service to store keys, secrets, and certificates with FIPS 140-2 Level 2 or 3.	Azure Key Vault basic concepts
RDP Short	Azure Virtual Desktop RDP Shortpath. This feature allows remote users to connect to the Virtual Desktop service from a private network.	Azure Virtual Desktop RDP Shortpath for managed networks
Reverse connect	A built-in security feature from Azure Virtual Desktop. Reverse connect guarantees that remote users receive only pixel streams and don't reach the host VMs.	Understanding Azure Virtual Desktop network connectivity

4. Application and data

The following table describes application and data services that are shown in the diagram.

[\[\]](#) [Expand table](#)

Label	Description	Documentation
Frontdoor + WAF	A content delivery network (CDN). Front Door combines multiple points of presence to deliver a better connection for users who access the service and adds WAF.	What is Azure Front Door?

Label	Description	Documentation
API Management	A service that delivers security for API calls and manages APIs across environments.	About API Management
PenTest	A set of best practices to execute a penetration test in your environment, including Azure resources.	Penetration testing
Storage SAS token	A shared access token using expiration policies to allow others to access your Azure storage account.	Grant limited access to Azure Storage resources using shared access signatures (SAS)
Private endpoint	Create a network interface and attach it to your storage account to configure it inside a private network on Azure.	Use private endpoints for Azure Storage
Storage firewall	Firewall that allows you to set a range of IP addresses that can access your storage account.	Configure Azure Storage firewalls and virtual networks
Encryption (Azure Storage)	Protects your storage account with encryption at rest.	Azure Storage encryption for data at rest
SQL audit	Tracks database events and writes them to an audit log in your Azure storage account.	Auditing for Azure SQL Database and Azure Synapse Analytics
Vulnerability assessment	Service that helps you discover, track, and remediate potential database vulnerabilities.	SQL vulnerability assessment helps you identify database vulnerabilities
Encryption (Azure SQL)	Transparent data encryption (TDE) helps protect Azure SQL database services by encrypting data at rest.	Transparent data encryption for SQL Database, SQL Managed Instance, and Azure Synapse Analytics

5. Identity

The following table describes identity services that are shown in the diagram.

Label	Description	Documentation
RBAC	Azure role-based access control (Azure RBAC) helps you manage access to Azure services by using granular permissions that are based on users' Microsoft Entra credentials.	What is Azure role-based access control (Azure RBAC)?
MFA	Multifactor authentication offers additional types of authentication beyond user names and passwords.	How it works: Microsoft Entra multifactor authentication
ID protection	Identity Protection, a security service from Microsoft Entra ID, analyzes trillions of signals per day to identify and protect users from threats.	What is Identity Protection?
PIM	Privileged Identity Management (PIM), a security service from Microsoft Entra ID. It helps you to provide superuser privileges temporarily for Microsoft Entra ID (for example, User Administrator) and Azure subscriptions (for example, Role Based Access Control Administrator or Key Vault Administrator).	What is Microsoft Entra Privileged Identity Management?
Cond Acc	Conditional Access is an intelligent security service that uses policies that you define for various conditions to block or grant access to users.	What is Conditional Access?

Components

- [Microsoft Entra ID](#) is an identity and access management service. In this architecture, it manages user identities and access to external resources such as Microsoft 365 and the Azure portal, and internal resources such as apps on your corporate intranet network.
- [Azure Virtual Network](#) is a networking service that enables secure communication between Azure resources, the internet, and on-premises networks. In this architecture, it provides the private network infrastructure that supports secure connectivity and isolation for workloads.

- [Azure Load Balancer](#) is a low-latency layer-4 load balancing service for UDP and TCP traffic. Load Balancer is a zone-redundant service that can handle millions of concurrent flows. In this architecture, it ensures high availability and scalability by distributing inbound and outbound traffic across resources in the virtual network.
- [Azure Virtual Machines](#) is an infrastructure as a service (IaaS) offering that provides scalable compute resources. In this architecture, VMs host workloads that require direct control over the operating system and security configurations.
- [Azure Kubernetes Service \(AKS\)](#) is a managed container orchestration service that simplifies deploying and managing Kubernetes clusters. In this architecture, AKS runs containerized applications and provides built-in features for security, governance, and continuous integration/continuous delivery (CI/CD).
- [Virtual Desktop](#) is a desktop and app virtualization service that delivers remote desktops from the cloud. In this architecture, it provides secure access to corporate desktops for remote users and includes built-in features like RDP Shortpath and reverse connect.
- [The Web Apps feature of App Service](#) hosts web applications, REST APIs, and mobile back ends. In this architecture, Web Apps hosts HTTP-based applications and provides security features like TLS and private endpoints. You can develop in your chosen language. Applications run and scale in both Windows and Linux-based environments.
- [Azure Storage](#) is a scalable and secure storage solution for various data types, including blobs, files, queues, and tables. In this architecture, it stores application and system data with encryption at rest and supports secure access via SAS tokens and private endpoints.
- [SQL Database](#) is a managed relational database service that automates patching, backups, and monitoring. In this architecture, it provides secure and compliant data storage via features like transparent data encryption, auditing, and vulnerability assessments.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Rudnei Oliveira](#) | Senior Azure Security Engineer

Other contributors:

- [Gary Moore](#) | Programmer/Writer
- [Andrew Nathan](#) | Senior Customer Engineering Manager

Next steps

Microsoft has more documentation that can help you secure your IT environment, and the following articles can be particularly helpful:

- [Security in the Microsoft Cloud Adoption Framework for Azure](#). The Cloud Adoption Framework provides security guidance for your cloud journey by clarifying the processes, best practices, models, and experience.
- [Microsoft Azure Well-Architected Framework](#). The Azure Well-Architected Framework is a set of guiding tenets that you can use to improve the quality of a workload. The framework is based on five pillars: reliability, security, cost optimization, operational excellence, and performance efficiency.
- [Microsoft Security Best Practices](#). Microsoft Security Best Practices (formerly known as the *Azure Security Compass* or *Microsoft Security Compass*) is a collection of best practices that provide clear, actionable guidance for security-related decisions.
- [Microsoft Cybersecurity Reference Architectures \(MCRA\)](#). MCRA is a compilation of various Microsoft security reference architectures.

In the following resources, you can find more information about the services, technologies, and terminologies that are mentioned in this article:

- [What are public, private, and hybrid clouds?](#)
- [Overview of the Azure Security Benchmark \(v3\)](#)
- [Embrace proactive security with Zero Trust](#)
- [Microsoft 365 subscription information](#)
- [Microsoft Defender XDR](#)

Related resources

For more information about this reference architecture, see the other articles in this series:

- Part 1: [Map threats to your IT environment](#)
- Part 3: [Build the second layer of defense with Microsoft Defender XDR Security services](#)
- Part 4: [Integration between Azure and Microsoft Defender XDR security services](#)

Build the second layer of defense with Microsoft Defender XDR Security services

Microsoft Defender for Office 365

Microsoft Defender for Cloud Apps

Microsoft Defender for Identity

Microsoft 365

Microsoft Endpoint Manager

Solution ideas

This article describes a solution idea. Your cloud architect can use this guidance to help visualize the major components for a typical implementation of this architecture. Use this article as a starting point to design a well-architected solution that aligns with your workload's specific requirements.

Many organizations operate in a hybrid environment, with resources hosted both on Azure and on-premises. Most Azure resources, such as virtual machines (VMs), Azure applications, and Microsoft Entra ID, can be secured using Azure's built-in security services.

In addition, organizations frequently subscribe to Microsoft 365 to provide users with applications like Word, Excel, PowerPoint, and Exchange Online. Microsoft 365 also offers security services that can be used to add an extra layer of protection to some of the most widely used Azure resources.

To effectively utilize Microsoft 365 security services, it's important to understand key terminology and the structure of Microsoft 365 services. This fourth article in a series of five explores these topics in greater detail, building on concepts covered in previous articles, particularly:

- [Map threats to your IT environment](#)
- [Build the first layer of defense with Azure Security services](#)

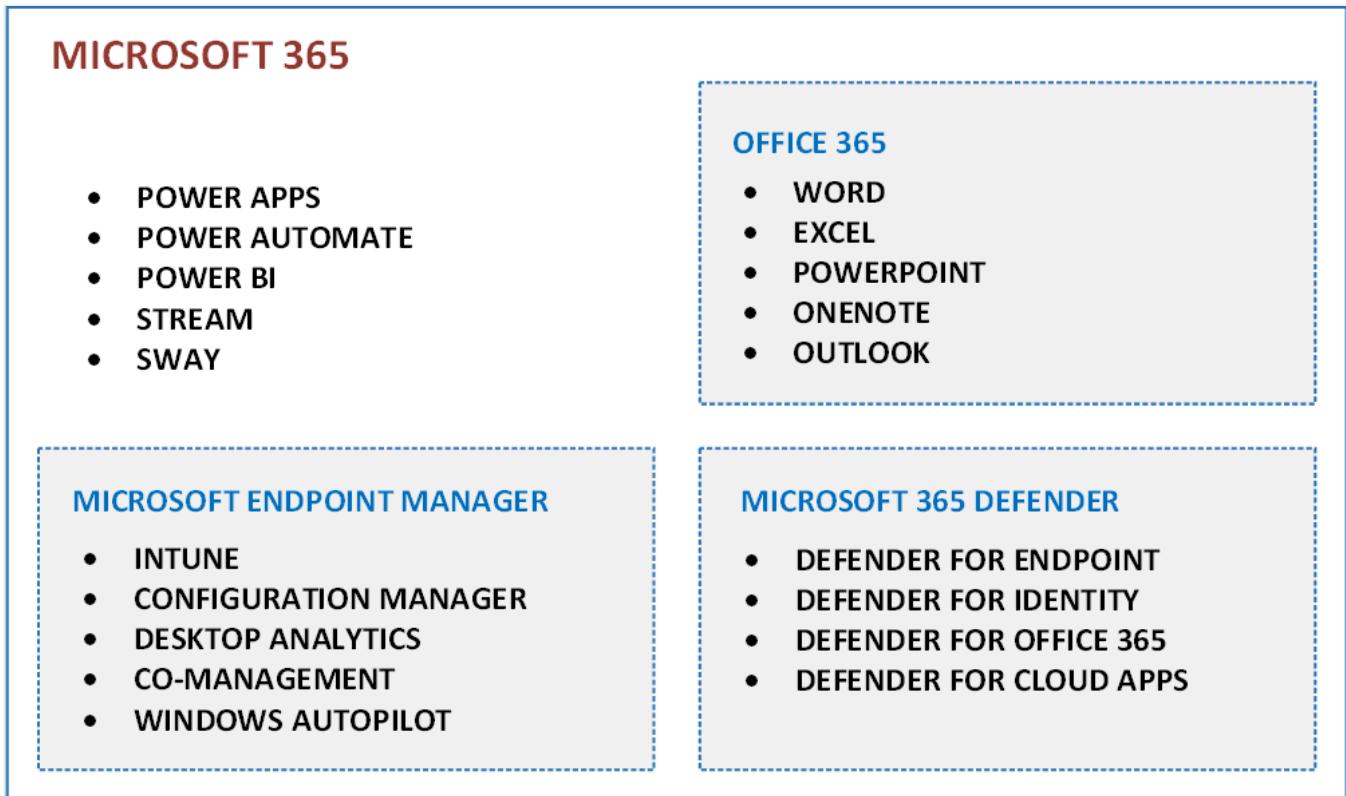
Microsoft 365 and Office 365 are cloud-based services designed to address your organization's needs for strong security, reliability, and enhanced user productivity. Microsoft 365 encompasses services such as Power Automate, Forms, Stream, Sway, and Office 365. Office 365 specifically includes the familiar suite of productivity applications. For more information about subscription options for these two services, see [Microsoft 365 and Office 365 plan options](#).

Depending on the license that you acquire for Microsoft 365, you can also get the security services for Microsoft 365. These security services are called Microsoft Defender XDR, which

provides multiple services:

- Microsoft Defender for Endpoint
- Microsoft Defender for Identity
- Microsoft Defender for Office
- Microsoft Defender for Cloud Apps
- "Microsoft Defender for Cloud Apps" accessed through "security.microsoft.com" is different from "Microsoft Defender for Cloud" that is another security solution accessed through "portal.azure.com".

The following diagram illustrates the relationship of solutions and main services that Microsoft 365 offers, though not all services are listed.



Potential use case

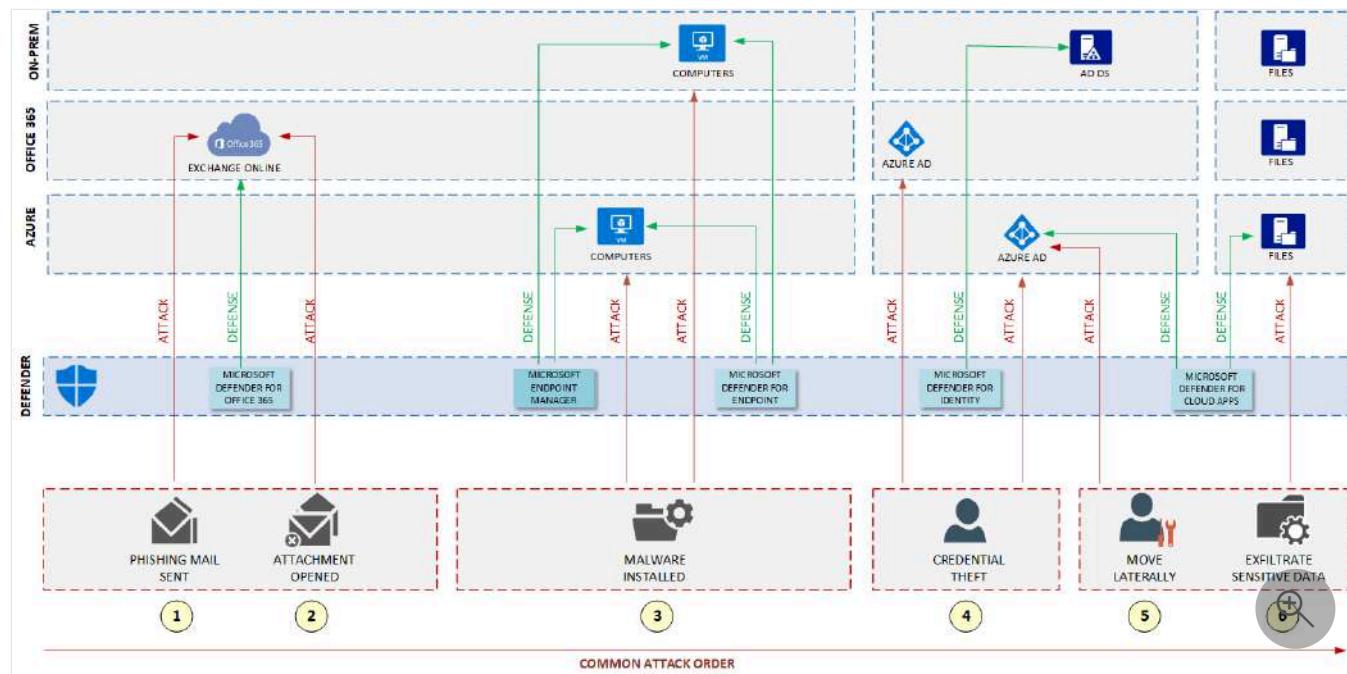
People often get confused about Microsoft 365 security services and their role in IT cybersecurity. A major cause of this confusion stems from the similarity in names, including some Azure security services like Microsoft Defender for Cloud (formerly Azure Security Center) and Defender for Cloud Apps (formerly Microsoft Cloud App Security).

However, the confusion goes beyond terminology. Some services provide similar protections but for different resources. For example, Defender for Identity and Azure Identity Protection both safeguard identity services, but Defender for Identity secures on-premises identities (via

Active Directory Domain Services and Kerberos authentication), while Azure Identity Protection secures cloud identities (via Microsoft Entra ID and OAuth authentication).

These examples highlight the importance of understanding how Microsoft 365 security services differ from Azure security services. By gaining this understanding, you can more effectively plan your security strategy in the Microsoft cloud while maintaining a strong security posture for your IT environment. This article aims to help you achieve that.

The following diagram presents a real-world use case for Microsoft Defender XDR security services. It shows the resources that need protection, the services running in the environment, and some potential threats. Microsoft Defender XDR services are positioned in the middle, defending the organization's resources from those threats.

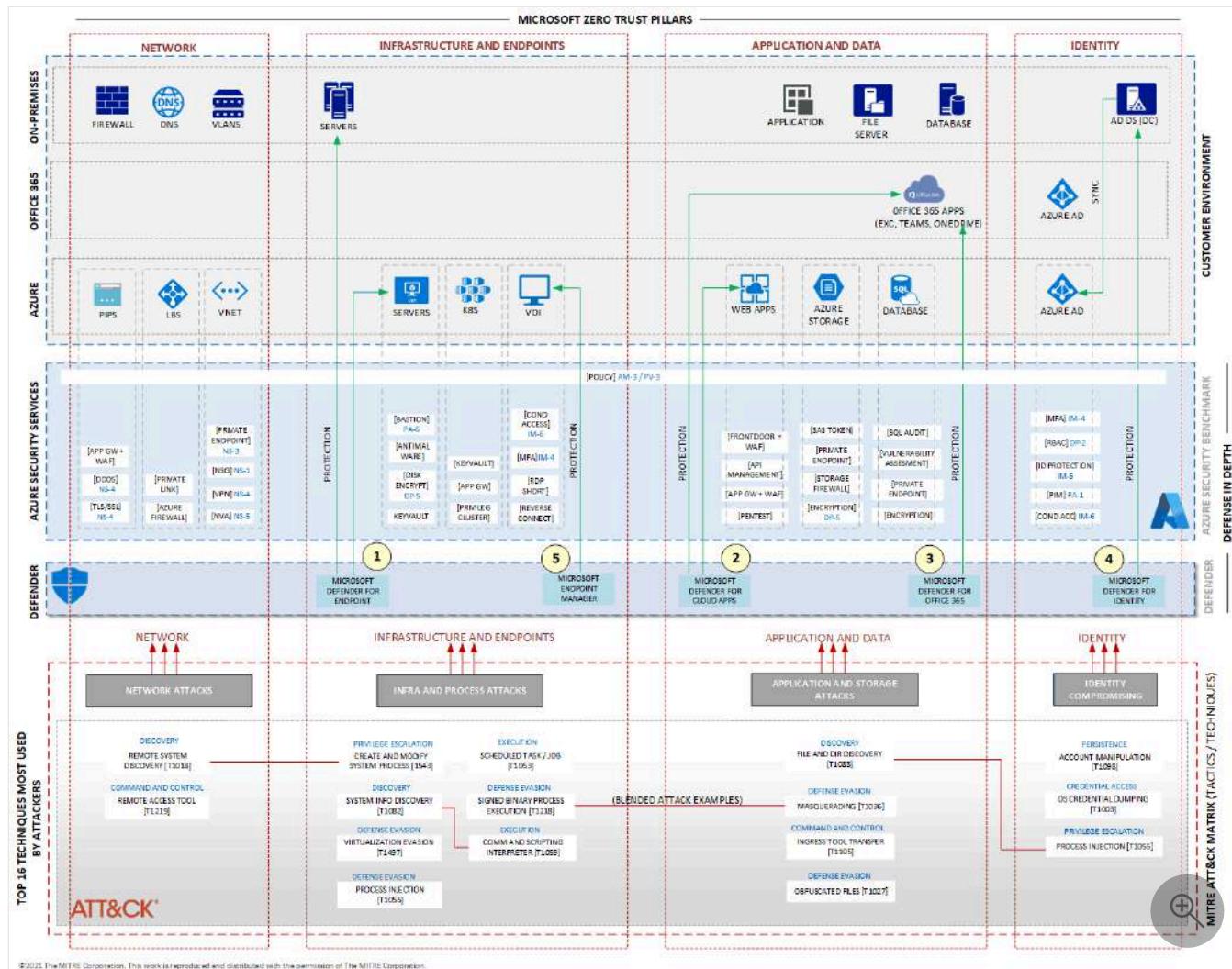


Architecture

Microsoft's Extended Detection and Response (XDR) solution, known as Microsoft Defender XDR, integrates multiple security tools and services to provide unified protection, detection, and response across endpoints, identities, email, applications, and cloud environments. It combines advanced threat intelligence, automation, and AI-driven analytics to detect and respond to sophisticated cyber threats in real-time, enabling security teams to quickly mitigate risks and reduce the impact of attacks. By consolidating security data from various sources, Microsoft Defender XDR helps organizations achieve comprehensive, streamlined defense across their entire IT infrastructure.

The following diagram shows a layer, labeled as **DEFENDER**, that represents the **Microsoft Defender XDR** security services. Adding these services to your IT environment helps you to

build better defense for your environment. The services in the Defender layer can work with Azure security services.



Download a [Visio file](#) of this architecture.

©2021 The MITRE Corporation. This work is reproduced and distributed with the permission of The MITRE Corporation.

Workflow

1. Microsoft Defender for Endpoint

Defender for Endpoint secures endpoints in your enterprise and is designed to help networks prevent, detect, investigate, and respond to advanced threats. It creates a layer of protection for VMs that run on Azure and on-premises. For more information about what it can protect, see [Microsoft Defender for Endpoint](#).

2. Microsoft Defender for Cloud Apps

Formerly known as Microsoft Cloud Application Security, Defender for Cloud Apps is a cloud access security broker (CASB) that supports multiple deployment modes. Those

modes include log collection, API connectors, and reverse proxy. It provides rich visibility, control over data travel, and sophisticated analytics to identify and combat cyberthreats across all your Microsoft and third-party cloud services. It provides protection and risk mitigation for Cloud Apps and even for some apps that run on-premises. It also provides a protection layer for users who access those apps. For more information, see [Microsoft Defender for Cloud Apps overview](#).

It's important to not confuse Defender for Cloud Apps with Microsoft Defender for Cloud, which provides recommendations and a score of the security posture of servers, apps, storage accounts, and other resources running in Azure, on-premises, and in other clouds. Defender for Cloud consolidates two previous services, Azure Security Center and Azure Defender.

3. Microsoft Defender for Office

Defender for Office 365 safeguards your organization against malicious threats that are posed by email messages, links (URLs), and collaboration tools. It provides protection for email and collaboration. Depending on the license, you're able to add post-breach investigation, hunting, and response, as well as automation and simulation (for training). For more information about licensing options, see [Microsoft Defender for Office 365 security overview](#).

4. Microsoft Defender for Identity

Defender for Identity is a cloud-based security solution that uses your on-premises Active Directory signals to identify, detect, and investigate advanced threats, compromised identities, and malicious insider actions that are directed at your organization. It protects Active Directory Domain Services (AD DS) that run on-premises. Even though this service runs on the cloud, it works to protect identities on-premises. Defender for Identity was formerly named Azure Advanced Threat Protection. For more information, see [What is Microsoft Defender for Identity?](#)

If you need protection for identities that are provided by Microsoft Entra ID and that runs natively on the cloud, consider Microsoft Entra ID Protection.

5. Intune (formerly part of the Microsoft Endpoint Manager)

Microsoft Intune is a cloud-based service that helps organizations manage and secure their devices, apps, and data. It allows IT administrators to control how company devices such as laptops, smartphones, and tablets are used, ensuring compliance with security policies. With Intune, you can enforce device configurations, deploy software, manage mobile applications, and protect corporate data by using features like Conditional Access and remote wipe. It is particularly useful for enabling secure remote work, managing both corporate-owned and

personal (BYOD) devices, and ensuring data security across diverse platforms like Windows, iOS, Android, and macOS.

Another service that was part of Endpoint Manager is the Configuration Manager, an on-premises management solution that allows you to manage client and server computers that are on your network, connected directly or via the internet. You can enable cloud functionality to integrate Configuration Manager with Intune, Microsoft Entra ID, Defender for Endpoint, and other cloud services. Use it to deploy apps, software updates, and operating systems. You can also monitor compliance, query for objects, act on clients in real time, and much more. To learn about all the services that are available, see [Microsoft Endpoint Manager overview](#).

Attack order of example threats

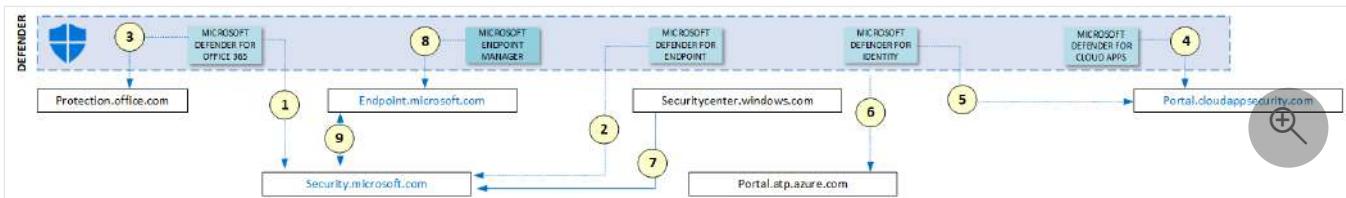
The threats named in the diagram follow a common attack order:

1. An attacker sends a phishing email with malware attached to it.
2. An end user opens the attached malware.
3. The malware installs in the back end without the user noticing.
4. The installed malware steals some users' credentials.
5. The attacker uses the credentials to gain access to sensitive accounts.
6. If the credentials provide access to an account that has elevated privilege, the attacker compromises additional systems.

The diagram also shows in the layer labeled as **DEFENDER** which Microsoft Defender XDR services can monitor and mitigate those attacks. This is an example of how Defender provides an additional layer of security that works with Azure security services to offer additional protection of the resources that are shown in the diagram. For more information about how potential attacks threaten your IT environment, see the second article in this series, [Map threats to your IT environment](#). For more information about Microsoft Defender XDR, see [Microsoft Defender XDR](#).

Access and manage Microsoft Defender XDR Security services

The following diagram shows which portals are currently available and their relationships with each other. In the time of the update for this articles, some of those portals might be already deprecated.



`Security.microsoft.com` is currently the most important portal available because it brings functionalities from Microsoft Defender for Office 365 (1), from Defender for Endpoint (2), from Defender for Office (3), Defender for Identity (5), Defender for Apps (4) and also for Microsoft Sentinel.

It is important to mention that Microsoft Sentinel has some features that still run only on the Azure Portal (`portal.azure.com`).

Lastly, `endpoint.microsoft.com` provides functionality mainly for Intune and Configuration Manager, but also for other services that are part of Endpoint Manager. Because `security.microsoft.com` and `endpoint.microsoft.com` deliver security protection for endpoints, they have many interactions between them (9) to offer a great security posture for your endpoints.

Components

The example architecture in this article uses the following Azure components:

- **Microsoft Entra ID** is a cloud-based identity and access management service. Microsoft Entra ID helps users access external and internal resources. In this architecture, Microsoft Entra ID authenticates users that access Microsoft 365, Azure, and software as a service (SaaS) applications. It serves as the identity foundation for threat detection and response.
- **Azure Virtual Network** is a networking service in Azure that enables secure communication between Azure resources, the internet, and on-premises networks. In this architecture, it provides the private network infrastructure that supports secure connectivity and segmentation of workloads that Microsoft Defender XDR protects.
- **Azure Load Balancer** is a high-performance, layer-4 load balancing service for Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) traffic. In this architecture, it ensures high availability and scalability for services that run in Azure by distributing traffic across VMs and containers.
- **Azure Virtual Machines** is an infrastructure-as-a-service (IaaS) offering that provides scalable compute resources. In this architecture, VMs host workloads that Microsoft Defender for Endpoint monitors and protects as part of the Microsoft Defender XDR solution.

- [Azure Kubernetes Service \(AKS\)](#) is a managed Kubernetes service for deploying and managing containerized applications. In this architecture, AKS runs containerized workloads that integrate into the Microsoft Defender XDR threat detection and response framework.
- [Azure Virtual Desktop](#) is a desktop and app virtualization service that provides secure remote access to cloud-hosted desktops. In this architecture, it supports remote users. Defender for Endpoint monitors Virtual Desktop to detect and respond to endpoint threats.
- The [Web Apps feature of Azure App Service](#) hosts web applications, REST APIs, and mobile back ends. You can develop in your chosen language. Applications run and scale with ease on both Windows and Linux-based environments. In this architecture, Web Apps hosts HTTP-based applications that are protected through integrated security features and monitored for threats.
- [Azure Storage](#) is a scalable and secure storage service for various data objects in the cloud, including object, blob, file, disk, queue, and table storage. Azure Storage encrypts all data written to an Azure storage account. It provides fine-grained control over access to your data. In this architecture, it stores application and system data and is protected by Defender for Cloud to ensure data integrity and access control.
- [Azure SQL Database](#) is a managed relational database engine that automates patching, backups, and monitoring. In this architecture, it stores structured data and benefits from built-in security features that align with Microsoft for Defender XDR threat protection capabilities.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Rudnei Oliveira](#) | Senior Customer Engineer

Other contributors:

- [Gary Moore](#) | Programmer/Writer
- [Andrew Nathan](#) | Senior Customer Engineering Manager

Next steps

- [Defend against threats with Microsoft 365](#)

- Detect and respond to cyber attacks with Microsoft Defender XDR
- Get started with Microsoft Defender XDR
- Implement threat intelligence in Microsoft 365
- Manage security with Microsoft 365
- Protect against malicious threats with Microsoft Defender for Office 365
- Protect on-premises identities with Microsoft Defender for Cloud for Identity

Related resources

For more information about this reference architecture, see the other articles in this series:

- Part 1: [Map threats to your IT environment](#)
- Part 2: [Build the first layer of defense with Azure Security services](#)
- Part 4: [Integration between Azure and Microsoft Defender XDR security services](#)

Integrate Azure and Microsoft Defender XDR security services

Microsoft Sentinel

Azure Monitor

Microsoft Defender for Cloud

Azure Log Analytics

Azure Network Watcher

Solution ideas

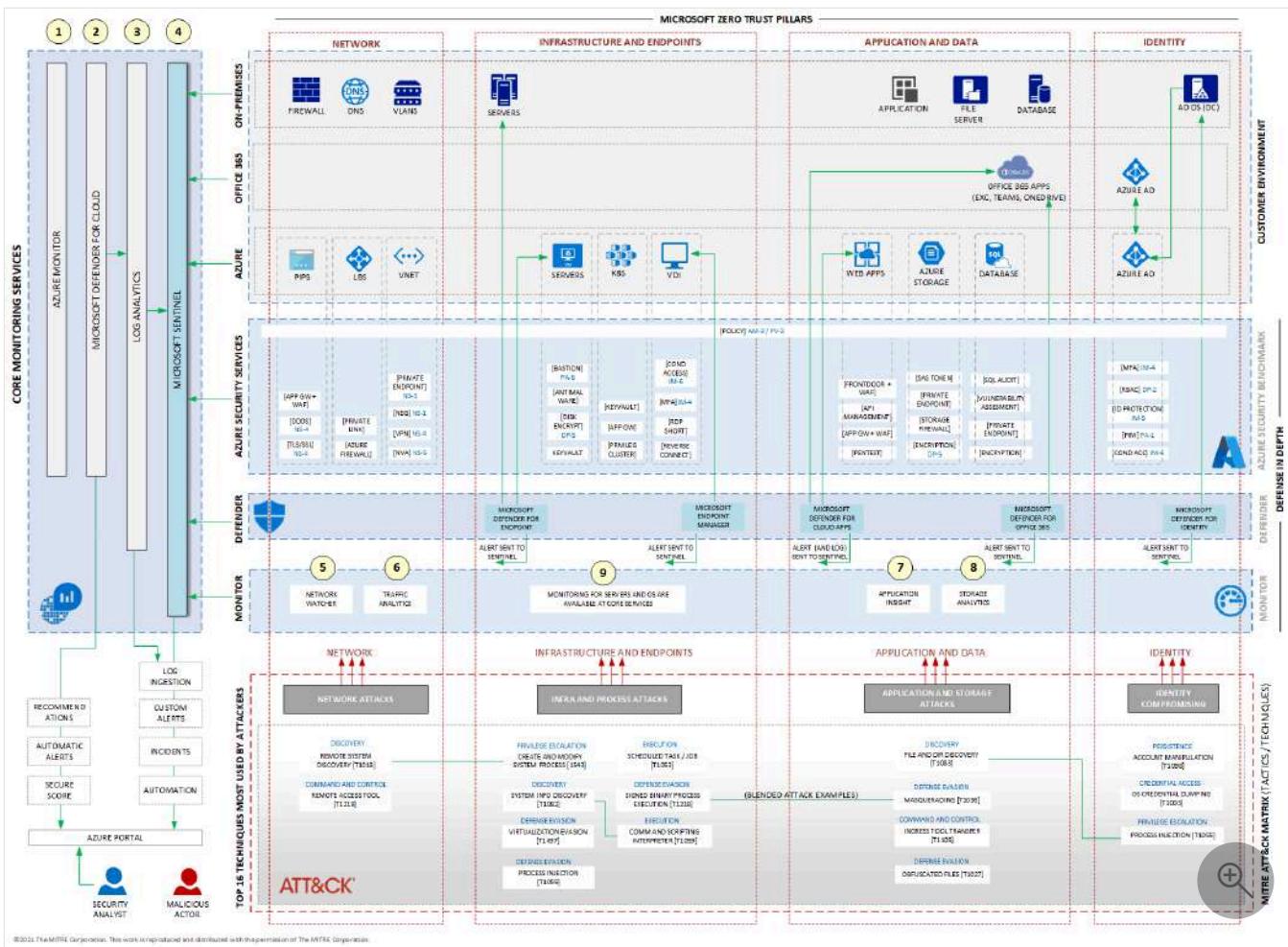
This article describes a solution idea. Your cloud architect can use this guidance to help visualize the major components for a typical implementation of this architecture. Use this article as a starting point to design a well-architected solution that aligns with your workload's specific requirements.

You can strengthen your organization's IT security posture by using the security features available in Microsoft 365 and Azure. This fifth and final article in the series explains how to integrate these security capabilities by using Microsoft Defender XDR and Azure monitoring services.

This article builds on the previous articles in the series:

1. [Map threats to your IT environment](#) describes methods to map examples of common threats, tactics, and techniques against an example of a hybrid IT environment that uses both on-premises and Microsoft cloud services.
2. [Build the first layer of defense with Azure Security services](#) maps an example of some Azure security services that create the first layer of defense to protect your Azure environment according to Azure Security Benchmark version 3.
3. [Build the second layer of defense with Microsoft Defender XDR Security services](#) describes an example of a series of attacks against your IT environment and how to add another layer of protection by using Microsoft Defender XDR.

Architecture



Download a [Visio file](#) of this architecture.

©2021 The MITRE Corporation. This work is reproduced and distributed with the permission of The MITRE Corporation.

This diagram shows a complete architecture reference. It includes an example of an IT environment, a set of example threats that are described according to their tactics (in blue), and their techniques (in the text box) according to the MITRE ATT&CK matrix. The MITRE ATT&CK matrix is covered in [Map threats to your IT environment](#).

The diagram highlights several important services. Some, like Azure Network Watcher and Application Insights, focus on capturing data from specific services. Others, like Log Analytics (also known as Azure Monitor Logs) and Microsoft Sentinel, serve as core services because they can collect, store, and analyze data from a wide range of services, whether related to networks, compute, or applications.

At the center of the diagram are two layers of security services and a layer dedicated to specific Azure monitoring services, all integrated through Azure Monitor (shown on the left side of the diagram). The key component of this integration is Microsoft Sentinel.

The diagram shows the following services in **Core Monitoring Services** and in the **Monitor** layer:

- Azure Monitor
- Log Analytics
- Microsoft Defender for Cloud
- Microsoft Sentinel
- Network Watcher
- Traffic Analytics (part of Network Watcher)
- Application Insights
- Storage Analytics

Workflow

1. **Azure Monitor** is the umbrella for many Azure monitoring services. It includes log management, metrics, and Application Insights, among others. It also provides a collection of dashboards that are ready for use and management of alerts. For more information, see [Azure Monitor overview](#).
2. **Microsoft Defender for Cloud** delivers recommendations for virtual machines (VMs), storage, applications, and other resources, that help an IT environment to be compliant with various regulatory standards, such as ISO and PCI. At the same time, Defender for Cloud offers a score for the security posture of systems that can help you track the security of your environment. Defender for Cloud also offers automatic alerts that are based on the logs that it collects and analyzes. Defender for Cloud was formerly known as Azure Security Center. For more information, see [Microsoft Defender for Cloud](#).
3. **Log Analytics** is one of the most important services. It's responsible for storing all the logs and alerts that are used to create alerts, insights, and incidents. Microsoft Sentinel works on top of Log Analytics. Basically, all data that Log Analytics ingests is available automatically to Microsoft Sentinel. Log Analytics is also known as Azure Monitor Logs. For more information, see [Overview of Log Analytics in Azure Monitor](#).
4. **Microsoft Sentinel** works like a façade for Log Analytics. While Log Analytics stores logs and alerts from various sources, Microsoft Sentinel offers APIs that help with ingestion of logs from various sources. Those sources include on-premises VMs, Azure VMs, alerts from Microsoft Defender XDR and other services. Microsoft Sentinel correlates the logs to provide insights about what is going on in your IT environment, avoiding false positives. Microsoft Sentinel is the core of security and monitoring for Microsoft cloud services. For more information about Microsoft Sentinel, see [What is Microsoft Sentinel?](#).

The preceding services in this list are core services that work throughout Azure, Office 365, and on-premises environments. The following services focus on specific resources:

5. **Network Watcher** provides tools to monitor, diagnose, view metrics, and enable or disable logs for resources in an Azure virtual network. For more information, see [What is Azure Network Watcher?](#).
6. **Traffic Analytics** is part of Network Watcher and works on top of logs from network security groups (NSGs). Traffic Analytics offers many dashboards that are capable of aggregating metrics from outbound and inbound connection in Azure Virtual Network. For more information, see [Traffic Analytics](#).
7. **Application Insights** focuses on applications and provides extensible performance management and monitoring for live web apps, including support for a wide variety of platform such as .NET, Node.js, Java, and Python. Application Insights is a feature of Azure Monitor. For more information, see [Application Insights overview](#).
8. **Azure Storage Analytics** performs logging and provides metrics for a storage account. You can use its data to trace requests, analyze usage trends, and diagnose issues with your storage account. For more information, see [Use Azure Storage analytics to collect logs and metrics data](#).

9. Because this architecture reference is based on [Microsoft Zero Trust](#), the services and components under **Infrastructure and Endpoint** don't have specific monitoring services. Azure Monitor logs and Defender for Cloud are the main services that collect, store, and analyze logs from VMs and others compute services.

The central component of this architecture is Microsoft Sentinel. It consolidates all the logs and alerts that are generated by Azure security services, Microsoft Defender XDR, and Azure Monitor. After Microsoft Sentinel is implemented and receiving logs and alerts from the sources outlined in this article, you need to map queries to those logs in order to gather insights and detect indicators of compromise (IOCs). When Microsoft Sentinel captures this information, you can either investigate it manually or trigger automated responses that you configure to mitigate or resolve incidents. Automated actions might include blocking a user in Microsoft Entra ID or blocking an IP address by using the firewall.

For more information about Microsoft Sentinel, see [Microsoft Sentinel documentation](#).

How to access security and monitoring services

The following list provides information about how to access each of the services that are presented in this article:

- **Azure security services.** You can access all the Azure security services that are mentioned in the diagrams in this series of articles by using [Azure portal](#). In the portal, use the search function to locate the services that you're interested in and access them.

- **Azure Monitor.** Azure Monitor is available in all Azure subscriptions. You can access it from a search for *monitor* in the [Azure portal](#).
- **Defender for Cloud.** Defender for Cloud is available to anyone who accesses the [Azure portal](#). In the portal, search for *Defender for Cloud*.
- **Log Analytics.** To access Log Analytics, you must first create the service in the portal, because it doesn't exist by default. In the [Azure portal](#), search for *Log Analytics workspace*, and then select **Create**. After creation, you're able to access the service.
- **Microsoft Sentinel.** Because Microsoft Sentinel works on top of Log Analytics, you must first create a Log Analytics workspace. Next, search for *sentinel* in the [Azure portal](#). Then create the service by choosing the workspace that you want to have behind Microsoft Sentinel.
- **Microsoft Defender for Endpoint.** Defender for Endpoint is part of Microsoft Defender XDR. Access the service through <https://security.microsoft.com>. This is a change from the previous URL, securitycenter.windows.com.
- **Microsoft Defender for Cloud Apps.** Defender for Cloud Apps is part of Microsoft 365. Access the service through <https://portal.cloudappsecurity.com>.
- **Microsoft Defender for Office 365.** Defender for Office 365 is part of Microsoft 365. Access the service through <https://security.microsoft.com>, the same portal used for Defender for Endpoint. (This is a change from the previous URL, protection.office.com.)
- **Microsoft Defender for Identity.** Defender for Identity is part of Microsoft 365. You access the service through <https://portal.atp.azure.com>. Although it's a cloud service, Defender for Identity is responsible for also protecting identity on on-premises systems.
- **Microsoft Endpoint Manager.** Endpoint Manager is the new name for Intune, Configuration Manager, and other services. Access it through <https://endpoint.microsoft.com>. To learn more about accessing the services that are provided by Microsoft Defender XDR and how each portal is related, see [Build the second layer of defense with Microsoft Defender XDR Security services](#).
- **Azure Network Watcher.** To access Azure Network Watcher, search for *watcher* in the [Azure portal](#).
- **Traffic Analytics.** Traffic Analytics is part of Network Watcher. You can access it from the menu on the left side in Network Watcher. It's a powerful network monitor that works based on your NSGs that are implemented on your individual network interfaces and subnets. Network Watcher requires collection of information from the NSGs. For

instructions on how to collect that information, see [Tutorial: Log network traffic to and from a virtual machine using the Azure portal](#).

- **Application Insight.** Application Insight is part of Azure Monitor. However, you must first create it for the application that you want to monitor. For some applications built on Azure, such as Web Apps, you can create Application Insight directly from the provisioning of Web Apps. To access it, search for *monitor* in the [Azure portal](#). In the **Monitor** page, select **Applications** in the menu on the left side.
- **Storage Analytics.** Azure Storage offers various types of storage under the same storage account technology. You can find blobs, files, tables, and queues on top of storage accounts. Storage analytics offers a broad range of metrics to use with those storage services. Access Storage Analytics from your Storage account in the [Azure portal](#), then select **Diagnostic settings** in the menu on the left side. Choose one log analytics workspace to send that information. Then you can access a dashboard from **Insights**. Everything in your storage account that's being monitored is represented in the menu.

Components

The example architecture in this article uses the following Azure components:

- [Microsoft Entra ID](#) is a cloud-based identity and access management service that helps users access external and internal resources. In this architecture, it authenticates users that access Microsoft 365, Azure, and software as a service (SaaS) applications, and supports identity-based threat detection and response.
- [Azure Virtual Network](#) is a networking service in Azure that enables secure communication between Azure resources, the internet, and on-premises networks. In this architecture, it provides the private network infrastructure for hosting workloads and collecting network-level telemetry.
- [Azure Load Balancer](#) is a high-performance layer-4 load balancing service for Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) traffic. In this architecture, it distributes traffic across VMs and services to ensure high availability and resilience.
- [Azure Virtual Machines](#) is an infrastructure as a service (IaaS) offering that provides scalable compute resources. In this architecture, VMs run workloads and applications that require full control over the operating system and environment.
- [Azure Kubernetes Service \(AKS\)](#) is a managed Kubernetes service for deploying and managing containerized applications. In this architecture, AKS orchestrates container

deployment and scaling, which supports microservices and continuous integration and continuous delivery (CI/CD) pipelines.

- [Azure Virtual Desktop](#) is a desktop and app virtualization service. In this architecture, it provides secure remote access to desktops and applications for distributed users.
- The [Web Apps feature of Azure App Service](#) hosts web applications, REST APIs, and mobile back ends. You can develop in your chosen language. Applications run and scale with ease on both Windows and Linux-based environments. In this architecture, Web Apps enables scalable and language-flexible deployment of web-based services.
- [Azure Storage](#) is scalable and secure storage for various data objects in the cloud, including object, blob, file, disk, queue, and table storage. In this architecture, it stores application data, logs, and backups with encryption and access control.
- [Azure SQL Database](#) is a managed relational database engine that automates upgrading, patching, backups, and monitoring. In this architecture, it provides secure, scalable, and compliant data storage for structured application data.

Solution details

Monitoring solutions on Azure might seem confusing at first, because Azure offers multiple monitoring services. However, each Azure monitoring service is important in the security and monitoring strategy that's described in this series. The articles in this series describe the various services and how to plan effective security for your IT environment.

1. [Map threats to your IT environment](#)
2. [Build the first layer of defense with Azure Security services](#)
3. [Build the second layer of defense with Microsoft Defender XDR Security services](#)

Potential use cases

This reference architecture provides a comprehensive view of Microsoft Cloud security services and demonstrates how to integrate them to achieve an optimal security posture.

Although you don't need to implement every security service shown, this example and the threat map illustrated in the architecture diagram can help you create your own threat map and plan your security strategy. Choose the Azure security services and Microsoft Defender XDR services that best suit your needs.

Cost optimization

Pricing for the Azure services that are presented in this series of articles is calculated in various ways. Some services are free of charge, some have a charge for each use, and some have a charge that is based on licensing. The best way to estimate the pricing for any of the Azure security services is to use the [Pricing calculator](#). In the calculator, search for a service that you're interested in, and then select it to get all the variables that determine the price for the service.

Microsoft Defender XDR security services work with licenses. For information about the licensing requirements, see [Microsoft Defender XDR prerequisites](#).

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Rudnei Oliveira](#) | Senior Azure Security Engineer

Other contributors:

- [Gary Moore](#) | Programmer/Writer
- [Andrew Nathan](#) | Senior Customer Engineering Manager

Next steps

- [Defend against threats with Microsoft 365](#)
- [Detect and respond to cyber attacks with Microsoft Defender XDR](#)
- [Get started with Microsoft Defender XDR](#)
- [Manage security with Microsoft 365](#)
- [Protect against malicious threats with Microsoft Defender for Office 365](#)
- [Protect on-premises identities with Microsoft Defender for Cloud for Identity](#)

Related resources

For more information about this reference architecture, see the other articles in this series:

- Part 1: [Map threats to your IT environment](#)
- Part 2: [Build the first layer of defense with Azure Security services](#)
- Part 3: [Build the second layer of defense with Microsoft Defender XDR Security services](#)

For related architectures on Azure Architecture Center, see the following article:

- [Implement a secure hybrid network](#)

Microsoft Sentinel automated responses

Microsoft Sentinel

Microsoft Entra ID

Azure Logic Apps

Solution ideas

This article describes a solution idea. Your cloud architect can use this guidance to help visualize the major components for a typical implementation of this architecture. Use this article as a starting point to design a well-architected solution that aligns with your workload's specific requirements.

Microsoft Sentinel is a scalable cloud-based solution for security information and event management (SIEM) and security orchestration, automation, and response (SOAR). It offers intelligent security analytics for organizations of all sizes and provides the following capabilities and more:

- Business attack detection
- Proactive hunting
- Automated incident response

Threat response in Microsoft Sentinel is managed via playbooks. When triggered by an alert or incident, a playbook runs a series of automated actions to counter the threat. You create these playbooks are by using Azure Logic Apps.

Microsoft Sentinel provides hundreds of ready-to-use playbooks, including playbooks for the following scenarios:

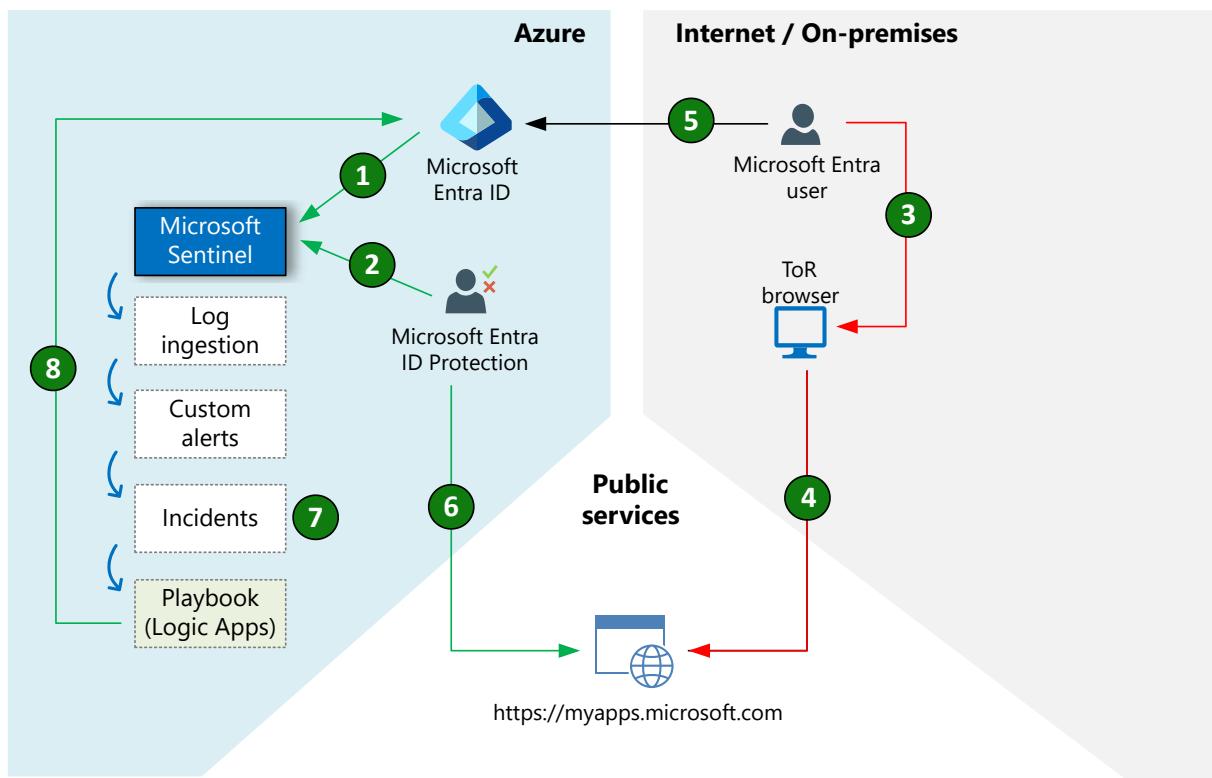
- Blocking a Microsoft Entra user
- Blocking a Microsoft Entra user based on rejection via email
- Posting a message in a Microsoft Teams channel about an incident or alert
- Posting a message on Slack
- Sending an email with incident or alert details
- Sending an email with a formatted incident report
- Determining whether a Microsoft Entra user is at risk
- Sending an adaptive card via Microsoft Teams to determine whether a user is compromised
- Isolating an endpoint via Microsoft Defender for Endpoint

This article includes an example of implementing a playbook that responds to a threat by blocking a Microsoft Entra user that's compromised by suspicious activity.

Potential use case

The techniques described in this article apply whenever you need to implement an automatic response to a detectable condition.

Architecture



Download a [Visio file](#) of this architecture.

Workflow

This workflow shows the steps to deploy the playbook. Make sure that the [Prerequisites](#) are satisfied before you start. For example, you need to choose a Microsoft Entra user.

1. Follow the steps in [Send logs to Azure Monitor](#) to configure Microsoft Entra ID to send audit logs to the Log Analytics workspace that's used with Microsoft Sentinel.

Note

This solution doesn't use the audit logs, but you can use them to investigate what happens when the user is blocked.

2. Microsoft Entra ID Protection generates the alerts that trigger the threat response playbook to run. To have Microsoft Sentinel collect the alerts, navigate to your Microsoft Sentinel instance and select **Data Connectors**. Search for **Microsoft Entra ID Protection** and enable the collecting of alerts. For more information about Identity Protection, see [What is Identity Protection?](#).
3. [Install the ToR browser](#) onto a computer or virtual machine (VM) that you can use without putting your IT security at risk.
4. Use the Tor Browser to sign in anonymously to My apps as the user that you selected for this solution. See [Anonymous IP address](#) for instructions on using the Tor Browser to simulate anonymous IP addresses.
5. Microsoft Entra authenticates the user.
6. Microsoft Entra ID Protection detects that the user used a ToR browser to sign in anonymously. This type of sign-in is suspicious activity that puts the user at risk. Identity Protection sends an alert to Microsoft Sentinel.
7. Configure Microsoft Sentinel to create an incident from the alert. For more information, see [Automatically create incidents from Microsoft security alerts](#). The Microsoft security analytics rule template to use is **Create incidents based on Microsoft Entra ID Protection alerts**.
8. When Microsoft Sentinel triggers an incident, the playbook responds with actions that block the user.

Components

- [Microsoft Sentinel](#) is a cloud-native SIEM and SOAR solution. It uses advanced AI and security analytics to detect and respond to threats across the enterprise. There are many playbooks on Microsoft Sentinel that you can use to automate your responses and protect your system.
- [Microsoft Entra ID](#) is a cloud-based directory and identity management service that combines core directory services, application access management, and identity protection into a single solution. It can synchronize with on-premises directories. The identity service provides single sign-on, multifactor authentication, and Conditional Access to guard against cybersecurity attacks. The solution shown in this article uses Microsoft Entra identity Protect to detect suspicious activity by a user.

- [Azure Logic Apps](#) is a serverless cloud service for creating and running automated workflows that integrate apps, data, services, and systems. Developers can use a visual designer to schedule and orchestrate common task workflows. Azure Logic Apps has [connectors](#) for many popular cloud services, on-premises products, and other software-as-a-service applications. In this solution, Azure Logic Apps runs the threat response playbook.

Considerations

- The Azure Well-Architected Framework is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Microsoft Azure Well-Architected Framework](#).
- Microsoft Sentinel offers more than 50 playbooks that are ready for use. You can find them on the **Playbook templates** tab of the [Microsoft Sentinel|Automation](#) page for your workspace.
- [GitHub](#) has a variety of Microsoft Sentinel playbooks that are built by the community.

Deploy this scenario

You can deploy this scenario by following the steps in [Workflow](#) after making sure that the [Prerequisites](#) are satisfied.

Prerequisites

- [Prepare the software and choose a test user](#)
- [Deploy the playbook](#)

Prepare the software and choose a test user

To implement and test the playbook, you need Azure and Microsoft Sentinel along with the following:

- A Microsoft Entra ID Protection license (Premium P2, E3, or E5).
- A Microsoft Entra user. You can use either an existing user or [create a new user](#). If you do create a new user, you can delete it when you're done using it.
- A computer or VM that can run a ToR browser. You'll use the browser to sign in to the My Apps portal as your Microsoft Entra user.

Deploy the playbook

To deploy a Microsoft Sentinel playbook, proceed as follows:

- If you don't have a Log Analytics workspace to use for this exercise, create a new one as follows:
 - Go to the [Microsoft Sentinel](#) main page, and select **+ Create** to get to the **Add Microsoft Sentinel to a workspace** page.
 - Select **Create a new workspace**. Follow the instructions to create the new workspace. After a short time, the workspace is created.
- At this point, you have a workspace, perhaps one that you just created. Use the following steps to see whether Microsoft Sentinel has been added to it, and to add it if not:
 - Go to the [Microsoft Sentinel](#) main page.
 - If Microsoft Sentinel has already been added to your workspace, the workspace appears in the displayed list. If it hasn't been added yet, add it as follows.
 - Select **+ Create** to get to the **Add Microsoft Sentinel to a workspace** page.
 - Select your workspace from the displayed list, and then select **Add** at the bottom of the page. After a short time, Microsoft Sentinel is added to your workspace.
- Create a playbook, as follows:
 - Go to the [Microsoft Sentinel](#) main page. Select your workspace. Select **Automation** from the left menu to get to the **Automation** page. This page has three tabs.
 - Select the **Playbook templates (Preview)** tab.
 - In the search field, enter **Block Microsoft Entra user - Incident**.
 - In the list of playbooks, select **Block Microsoft Entra user - Incident** and then select **Create playbook** in the bottom right corner to get to the **Create playback** page.
 - On the **Create playbook** page, do the following:
 - Select values for **Subscription**, **Resource group**, and **Region** from the lists.
 - Enter a value for **Playbook name** if you don't want to use the default name that appears.
 - If you want, select **Enable diagnostics logs in Log Analytics** to enable logs.
 - Select **Next: Connections >** to go to the **Connections** tab of **Create playbook**.
 - Choose how to authenticate within the playbook's components. Authentication is required for:
 - Microsoft Entra ID
 - Microsoft Sentinel
 - Office 365 Outlook

 **Note**

You can authenticate the resources during playbook customization under the logic app resource if you wish to enable later. To authenticate the above resources

at this point, you need permissions to update a user on Microsoft Entra ID, and the user must have access to an email mailbox and must be able to send emails.

- Select **Next: Review and create** > to get to the **Review and create** tab of **Create playbook**.
- Select **Create and continue to designer** to create the playbook and access the **Logic app designer** page.

For more information about building logic apps, see [What is Azure Logic Apps](#) and [Quickstart: Create and manage logic app workflow definitions](#).

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Rudnei Oliveira](#) | Senior Azure Security Engineer

Other contributors:

- [Andrew Nathan](#) | Senior Customer Engineering Manager
- [Lavanya Kasturi](#) | Technical Writer

Related content

- [Overview of Azure Cloud Services?](#)
- [What is Microsoft Sentinel?](#)
- [Security orchestration, automation, and response \(SOAR\) in Microsoft Sentinel.](#)
- [Automate threat response with playbooks in Microsoft Sentinel](#)
- [What is Microsoft Entra ID?](#)
- [What is Identity Protection?](#)
- [Simulating risk detections in Identity Protection](#)
- [What is Azure Logic Apps?](#)
- [Tutorial: Create automated approval-based workflows by using Azure Logic Apps](#)
- [Introduction to Microsoft Sentinel](#)

Related resource

- [Security architecture design](#)

Storage architecture design

Article • 10/04/2024

The Azure Storage platform is the Microsoft cloud storage solution for modern data storage scenarios.

The Azure Storage platform includes the following data services:

- [Azure Blob Storage](#) : A massively scalable object store for text and binary data. Also includes support for big data analytics through Azure Data Lake Storage Gen2.
- [Azure Files](#) : Managed file shares for cloud or on-premises deployments.
- [Azure Queue Storage](#) : A messaging store for reliable messaging between application components.
- [Azure Table Storage](#) : A NoSQL store for schemaless storage of structured data.
- [Azure Disk Storage](#) : Block-level storage volumes for Azure VMs.
- [Azure NetApp Files](#) : An Azure native, first-party, enterprise-class, high-performance file storage service.

Introduction to storage on Azure

If you're new to storage on Azure, the best way to learn more is [Microsoft Learn training](#). This free online platform provides interactive learning for Microsoft products and more. Check out the [Store data in Azure](#) learning path.

Path to production

- Choose the storage approach that best meets your needs and then create an account. For more information, see [Storage account overview](#). For information about Azure NetApp Files, see [Storage hierarchy of Azure NetApp Files](#).
- Be sure you understand security and reliability. See these articles:
 - [Azure Storage encryption for data at rest](#)
 - [Use private endpoints - Azure Storage](#)
 - [Data redundancy - Azure Storage](#)
 - [Disaster recovery and storage account failover - Azure Storage](#)
 - [Understand data encryption in Azure NetApp Files](#)
 - [Understand data protection and disaster recovery options in Azure NetApp Files](#)

- For information about migrating existing data, see the [Azure Storage migration guide](#).

Best practices

Depending on the storage technology you use, see the following best practices resources:

- [Performance and scalability checklist for Blob Storage](#)
- [Best practices for using Azure Data Lake Storage Gen2](#)
- [Planning for an Azure Files deployment](#)
- [Performance and scalability checklist for Queue Storage](#)
- [Azure Storage table design patterns](#)
- [Solution architectures using Azure NetApp Files](#)
- [Performance considerations for Azure NetApp Files](#)

Blob Storage

See the following guides for information about Blob Storage:

- [Authorize access to blobs using Microsoft Entra ID](#)
- [Security recommendations for Blob Storage](#)

Azure Data Lake Storage

See the following guides for information about Data Lake Storage:

- [Best practices for using Azure Data Lake Storage Gen2](#)
- [Azure Policy Regulatory Compliance controls for Azure Data Lake Storage Gen1](#)

Azure Files

See the following guides for information about Azure Files:

- [Planning for an Azure Files deployment](#)
- [Overview of Azure Files identity-based authentication options for SMB access](#)
- [Disaster recovery and storage account failover](#)
- [About Azure file share backup](#)

Azure NetApp Files

See the following guides for information about Azure NetApp Files:

- [Solution architectures using Azure NetApp Files](#)
- [Storage hierarchy of Azure NetApp Files](#)
- [Service levels for Azure NetApp Files](#)
- [Understand data protection and disaster recovery options in Azure NetApp Files](#)
- [Guidelines for Azure NetApp Files network planning](#)
- [Quickstart: Set up Azure NetApp Files and NFS volume](#)

Queue Storage

See the following guides for information about Queue Storage:

- [Authorize access to queues using Microsoft Entra ID](#)
- [Performance and scalability checklist for Queue Storage](#)

Table Storage

See the following guides for information about Table Storage:

- [Authorize access to tables using Microsoft Entra ID \(preview\)](#)
- [Performance and scalability checklist for Table storage](#)
- [Design scalable and performant tables](#)
- [Design for querying](#)

Azure Disk Storage

See the following guides for information about Azure managed disks:

- [Server-side encryption of Azure Disk Storage](#)
- [Azure Disk Encryption for Windows VMs](#)
- [Azure premium storage: design for high performance](#)
- [Scalability and performance targets for VM disks](#)

Stay current with storage

Get the [latest updates on Azure Storage products and features](#).

Additional resources

To plan for your storage needs, see [Review your storage options](#).

Example solutions

Here are a few sample implementations of storage on Azure:

- [Using Azure file shares in a hybrid environment](#)
- [Azure files accessed on-premises and secured by AD DS](#)
- [Enterprise file shares with disaster recovery](#)
- [Hybrid file services](#)
- [Medical data storage solutions](#)
- [HPC media rendering](#)

[See more storage examples in the Azure Architecture Center.](#)

AWS or Google Cloud professionals

These articles provide service mapping and comparison between Azure and other cloud services. They can help you ramp up quickly on Azure.

- [Compare AWS and Azure Storage services](#)
- [Google Cloud to Azure services comparison - Storage](#)

Feedback

Was this page helpful?

 Yes

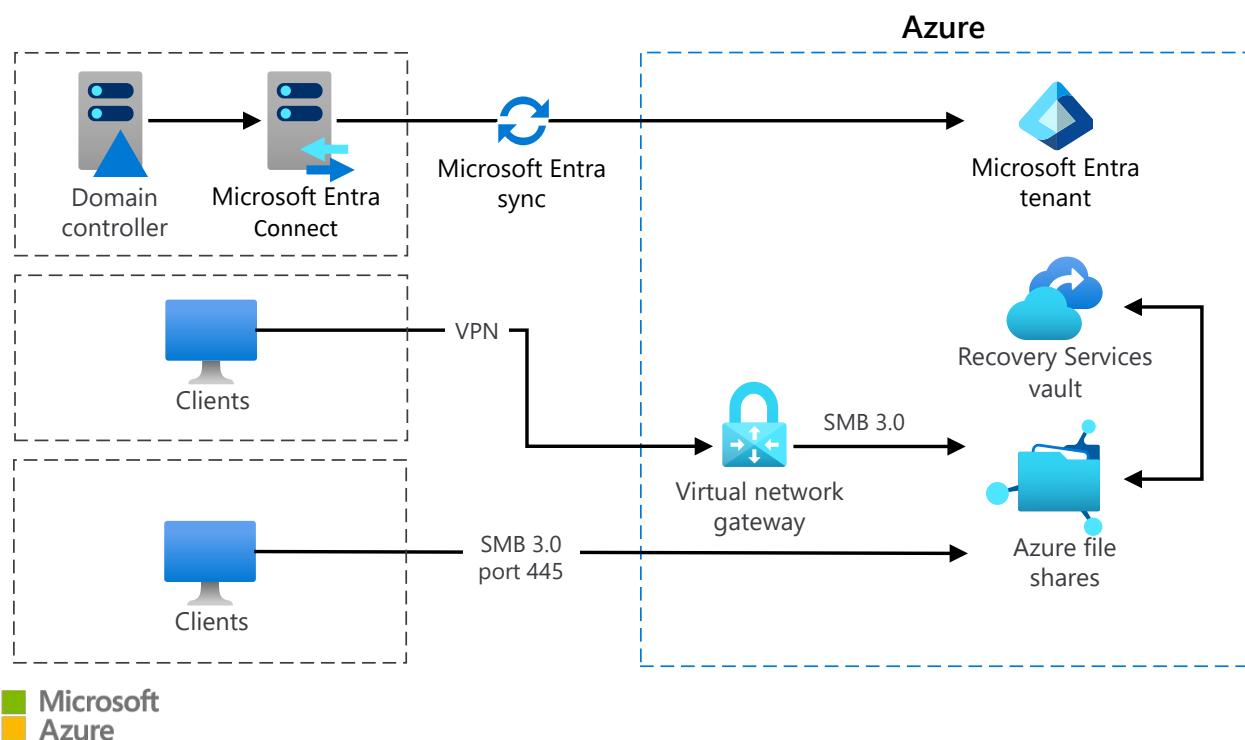
 No

Use Azure file shares in a hybrid environment

Microsoft Entra ID Azure Files

This architecture shows how to include Azure file shares in your hybrid environment. Azure file shares are used as serverless file shares. By integrating them with Active Directory Domain Services (AD DS), you can control and limit access to AD DS users. Azure file shares then can replace traditional file servers.

Architecture



Download a [Visio file](#) of this architecture.

Components

The architecture consists of the following components:

- **Microsoft Entra ID** is an enterprise identity service that provides features to protect against cybersecurity threats. In this architecture, it serves as the cloud-based directory that stores synchronized objects from on-premises AD DS and authenticates users that access Azure file shares.
- **AD DS** is an on-premises identity and directory service. In this architecture, it authenticates domain-joined clients and integrates with Azure Files to enforce access

controls by using Windows access control lists (ACLs). The AD DS directory synchronizes with Microsoft Entra ID to authenticate on-premises users.

- [Microsoft Entra Connect Sync](#) is a synchronization service that runs on an on-premises server. In this architecture, it ensures identity consistency by syncing AD DS objects to Microsoft Entra ID.
- [Azure VPN Gateway](#) is a networking service that sends encrypted traffic between Azure and on-premises networks over the public internet. In this architecture, this optional component provides secure connectivity for accessing Azure file shares when Server Message Block (SMB) port 445 is blocked by internet service providers (ISPs) or firewalls.
- [Azure Files](#) provides managed cloud-based file shares that can be accessed via SMB, Network File System (NFS), and HTTP protocols. In this architecture, it replaces traditional file servers by providing scalable, secure, and highly available storage integrated with AD DS authentication. File shares are deployed into Azure storage accounts.
- [Azure Recovery Services](#) is a suite of services designed to support data protection, backup, and disaster recovery. In this architecture, an optional Recovery Services vault protects Azure file shares by enabling backup and recovery through incremental share snapshots.
- **Clients** are user computing devices, such as desktops, laptops, or mobile devices, that access resources within the network. In this architecture, clients refer to AD DS domain-joined computers for users. These clients access Azure file shares by using their existing credentials, which maintains a familiar experience while taking advantage of cloud-based storage.

Potential use cases

Typical uses for this architecture include:

- **Replace or supplement on-premises file servers.** Azure Files can completely replace or supplement traditional on-premises file servers or network-attached storage devices. With Azure file shares and AD DS authentication, you can migrate data to Azure Files. This migration can take the advantage of high availability and scalability while minimizing client changes.
- **Lift and shift.** Azure Files supports "lift and shift" of applications that expect a file share to store application or user data to the cloud.
- **Backup and disaster recovery.** You can use Azure Files as storage for backups or for disaster recovery to improve business continuity. You can use Azure Files to back up your data from existing file servers while preserving configured Windows discretionary access control lists. Data that's stored on Azure file shares isn't affected by disasters that might affect on-premises locations.
- **Azure File Sync.** With Azure File Sync, Azure file shares can replicate to Windows Server, either on-premises or in the cloud. This replication improves performance and distributes

caching of data to where it's being used.

Recommendations

The following recommendations apply for most scenarios. Follow these recommendations unless you have a specific requirement that overrides them.

Use general-purpose v2 (GPv2) or FileStorage storage accounts for Azure file shares

You can create an Azure file share in various storage accounts. Although general-purpose v1 (GPv1) and classic storage accounts can contain Azure file shares, most new features of Azure Files are available only in GPv2 and FileStorage storage accounts. While an Azure file share stores GPv2 storage accounts data on hard disk drive-based (HDD-based) hardware, it stores FileStorage storage accounts data on solid-state drive-based (SSD-based) hardware. For more information, see [Create an Azure file share](#).

Create Azure file shares in storage accounts that contain only Azure file shares

Storage accounts allow you to use different storage services in the same storage account. These storage services include Azure file shares, blob containers, and tables. All storage services in a single storage account share the same storage account limits. Mixing storage services in the same storage account make it more difficult to troubleshoot performance issues.

 **Note**

Deploy each Azure file share in its own separate storage account, if possible. If multiple Azure file shares are deployed into the same storage account, they all share the storage account limits.

Use premium file shares for workloads that require high throughput

Premium file shares are deployed to FileStorage storage accounts and are stored on solid-state drive-based (SSD-based) hardware. This setup makes them suitable for storing and accessing data that requires consistent performance, high throughput, and low latency. (For example, these premium file shares work well with databases.) You can store other workloads that are

less sensitive to performance variability on standard file shares. These workload types include general-purpose file shares and dev/test environments. For more information, see [How to create an Azure file share](#).

Always require encryption when accessing SMB Azure file shares

Always use encryption in transit when accessing data in SMB Azure file shares. Encryption in transit is enabled by default. Azure Files will only allow the connection if it's made with a protocol that uses encryption, such as SMB 3.0. Clients that don't support SMB 3.0 will be unable to mount the Azure file share if encryption in transit is required.

Use VPN if port that SMB uses (port 445) is blocked

Many internet service providers block **Transmission Control Protocol (TCP) port 445**, which is used to access Azure file shares. If unblocking TCP port 445 isn't an option, you can access Azure file shares over an ExpressRoute or virtual private network (VPN) connection (site-to-site or point-to-site) to avoid traffic blocking. For more information, see [Configure a Point-to-Site \(P2S\) VPN on Windows for use with Azure Files](#) and [Configure a Site-to-Site VPN for use with Azure Files](#).

Consider using Azure File Sync with Azure file shares

The Azure File Sync service allows you to cache Azure file shares on an on-premises Windows Server file server. When you enable cloud tiering, File Sync helps ensure a file server always has free available space, even as it makes more files available than a file server could store locally. If you have on-premises Windows Server file servers, consider integrating file servers with Azure file shares by using Azure File Sync. For more information, see [Planning for an Azure File Sync deployment](#).

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that can be used to improve the quality of a workload. For more information, see [Microsoft Azure Well-Architected Framework](#).

Scalability

- Azure file share size is limited to 100 tebibytes (TiB). There's no minimum file share size and no limit on the number of Azure file shares.

- Maximum size of a file in a file share is 1 TiB, and there's no limit on the number of files in a file share.
- IOPS and throughput limits are per Azure storage account and are shared between Azure file shares in the same storage account.

For more information, see [Azure Files scalability and performance targets](#).

Availability

! Note

An Azure storage account is the parent resource for Azure file shares. Azure file share has the level of redundancy that's provided by the storage account that contains the share.

- Azure file shares currently support the following data redundancy options:
 - **Locally redundant storage (LRS)**. Data is copied synchronously three times within a single physical location in the primary region. This practice protects against loss of data because of hardware faults, such as a bad disk drive.
 - **Zone-redundant storage (ZRS)**. Data is copied synchronously across three Azure availability zones in the primary region. Availability zones are unique physical locations within an Azure region. Each zone consists of one or more datacenters equipped with independent power, cooling, and networking.
 - **Geo-redundant storage (GRS)**. Data is copied synchronously three times within a single physical location in the primary region using LRS. Your data is then copied asynchronously to a single physical location in the secondary region. Geo-redundant storage provides six copies of your data spread between two Azure regions.
 - **Geo-zone-redundant storage (GZRS)**. Data is copied synchronously across three Azure availability zones in the primary region using ZRS. Your data is then copied asynchronously to a single physical location in the secondary region.
- Premium file shares can be stored in locally redundant storage (LRS) and zone redundant storage (ZRS) only. Standard file shares can be stored in LRS, ZRS, geo-redundant storage (GRS), and geo-zone-redundant storage (GZRS). For more information, see [Planning for an Azure Files deployment](#) and [Azure Storage redundancy](#).
- Azure Files is a cloud service, and as with all cloud services, you must have internet connectivity to access Azure file shares. A redundant internet connection solution is highly recommended to avoid disruptions.

Manageability

- You can manage Azure file shares by using the same tools as any other Azure service. These tools include Azure portal, Azure Command-Line Interface, and Azure PowerShell.
- Azure file shares enforce standard Windows file permissions. You can configure directory or file-level permissions by mounting an Azure file share and configuring permissions using File Explorer, Windows `icacls.exe` command, or the `Set-Acl` Windows PowerShell cmdlet.
- You can use Azure file share snapshot for creating a point-in-time, read-only copy of the Azure file share data. You create a share snapshot at the file share level. You can then restore individual files in the Azure portal or in File Explorer, where you can also restore a whole share. You can have up to 200 snapshots per share, which enables you to restore files to different point-in time versions. If you delete a share, its snapshots are also deleted. Share snapshots are incremental. Only the data that has changed after your most recent share snapshot is saved. This practice minimizes the time required to create the share snapshot and saves on storage costs. Azure file share snapshots are also used when you protect Azure file shares with Azure Backup. For more information, see [Overview of share snapshots for Azure Files](#).
- You can prevent accidental deletion of Azure file shares by enabling soft delete for file shares. If you delete a file share when a soft delete is enabled, file share transitions to a soft deleted state instead of being permanently erased. You can configure the amount of time soft deleted data is recoverable before it's permanently deleted and restore the share anytime during this retention period. For more information, see [Enable soft delete on Azure file shares](#).

 **Note**

Azure Backup enables soft delete for all file shares in the storage account when you configure backup for the first Azure file share in the respective storage account.

 **Note**

Both standard and premium file shares are billed on used capacity when soft deleted, rather than provisioned capacity.

Security

Security provides assurances against deliberate attacks and the abuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

- Use AD DS authentication over SMB for accessing Azure file shares. This setup provides the same single sign-on (SSO) experience when accessing Azure file shares as accessing

on-premises file shares. For more information, see [How it works](#) and feature [enablement steps](#). Your client needs to be domain joined to AD DS, because the authentication is still done by the AD DS domain controller. Also, you need to assign both share level and file/directory level permissions to get access to the data. [Share level permission assignment](#) goes through Azure role-based access control (Azure RBAC) model. [File/directory level permission](#) is managed as Windows ACLs.

 **Note**

Access to Azure file shares is always authenticated. Azure file shares don't support anonymous access. Besides identity-based authentication over SMB, users can authenticate to Azure file share also by using storage access key and Shared Access Signature.

- All data that's stored on Azure file share is encrypted at rest using Azure storage service encryption (SSE). SSE works similarly to BitLocker Drive Encryption on Windows, where data is encrypted beneath the file system level. By default, data stored in Azure Files is encrypted with Microsoft-managed keys. With Microsoft-managed keys, Microsoft maintains the keys to encrypt/decrypt the data and manages rotating them regularly. You can also choose to manage your own keys, which gives you control over the rotation process.
- All Azure storage accounts have encryption in transit enabled by default. This setup means that all communication with Azure file shares is encrypted. Clients that don't support encryption can't connect to Azure file shares. If you disable encryption in transit, clients that run older operating systems, such as Windows Server 2008 R2 or older Linux, can also connect. In such instances, data isn't encrypted in transit from Azure file shares.
- By default, clients can connect to Azure file share from anywhere. To limit the networks from which clients can connect to Azure file shares, configure the Firewall, virtual networks, and private endpoint connections. For more information, see [Configure Azure Storage firewalls and virtual networks](#) and [Configuring Azure Files network endpoints](#).

Cost Optimization

Cost Optimization is about looking at ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

- Azure Files has two storage tiers and two pricing models:

- **Standard storage:** Uses HDD-based storage. There's no minimum file share size, and you pay only for used storage space. Also, you pay for file operations, such as enumerating a directory or reading a file.
- **Premium storage:** Uses SSD-based storage. The minimum size for a premium file share is 100 gibibytes, and you pay per provisioned storage space. When using premium storage, all file operations are free.
- Extra costs are associated with file share snapshots and outbound data transfers. (When you transfer data from Azure file shares, inbound data transfer is free.) Data transfer costs depend on the amount of transferred data and the stock keeping unit (SKU) of your virtual network gateway, if you use one. For more information about costs, see [Azure Files Pricing](#) and [Azure Pricing calculator](#). The actual cost varies by Azure region and your individual contract. For more information about pricing, contact a Microsoft sales representative.

Next steps

Learn more about the component technologies:

- [How to create an Azure file share](#) for instructions on getting started with an SMB share.
- [How to create an NFS share](#) for instructions on getting started with an NFS mount share.

Related resources

Explore related architectures:

- [Azure enterprise cloud file share](#)
- [Hybrid file services](#)
- [Azure Virtual Desktop for the enterprise](#)

Azure Files accessed from on-premises and secured by AD DS in a private network

Azure Virtual Network

Azure ExpressRoute

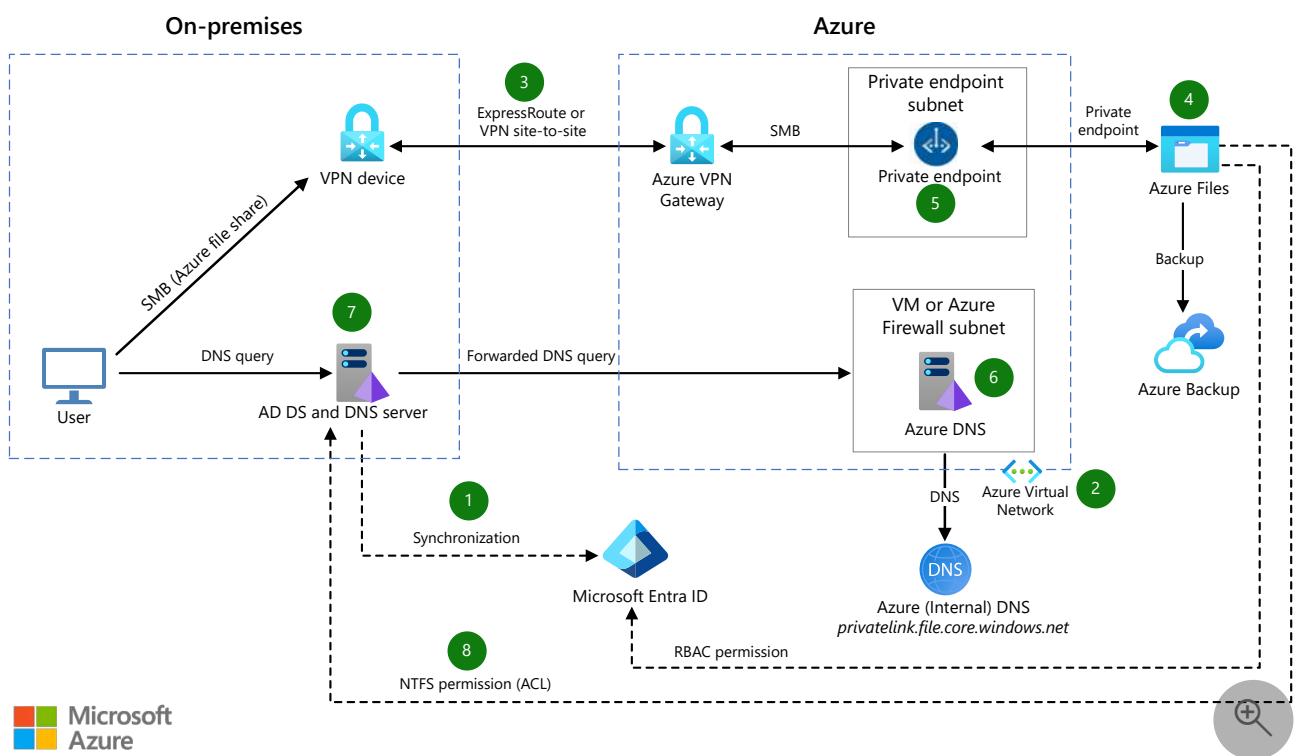
Azure Storage Accounts

Azure Files

Azure DNS

This architecture demonstrates one way to provide file shares in the cloud to on-premises users and applications that access files on Windows Server through a private endpoint.

Architecture



Download a [Visio file](#) of this architecture.

Workflow

1. This solution synchronizes the on-premises AD DS and the cloud-based Microsoft Entra ID. Synchronizing makes users more productive by providing a common identity for accessing both cloud and on-premises resources.

Microsoft Entra Connect is the on-premises Microsoft application that does the synchronizing. For more information about Microsoft Entra Connect, see [What is Microsoft Entra Connect](#).

[Microsoft Entra Connect?](#) and [Microsoft Entra Connect Sync: Understand and customize synchronization.](#)

2. Azure Virtual Network provides a virtual network in the cloud. For this solution, it has at least two subnets, one for Azure DNS, and one for a private endpoint to access the file share.
3. Either VPN or Azure ExpressRoute provides secure connections between the on-premises network and the virtual network in the cloud. If you use VPN, create a gateway by using Azure VPN Gateway. If you use ExpressRoute, create an ExpressRoute virtual network gateway. For more information, see [What is VPN Gateway?](#) and [About ExpressRoute virtual network gateways](#).
4. Azure Files provides a file share in the cloud. This requires an Azure Storage account. For more information about file shares, see [What is Azure Files?](#).
5. A private endpoint provides access to the file share. A private endpoint is like a network interface card (NIC) inside a subnet that attaches to an Azure service. In this case, the service is the file share. For more information about private endpoints, see [Use private endpoints for Azure Storage](#).
6. The on-premises DNS server resolves IP addresses. However, Azure DNS resolves the Azure file share Fully Qualified Domain Name (FQDN). All DNS queries to Azure DNS originate from the virtual network. There's a DNS proxy inside the virtual network to route these queries to Azure DNS. For more information, see [On-premises workloads using a DNS forwarder](#).

You can provide the DNS proxy on a Windows or Linux server, or you can use Azure Firewall. For information on the Azure Firewall option, which has the advantage that you don't have to manage a virtual machine, see [Azure Firewall DNS settings](#).
7. The on-premises custom DNS is configured to forward DNS traffic to Azure DNS via a conditional forwarder. Information on conditional forwarding is also found in [On-premises workloads using a DNS forwarder](#).
8. The on-premises AD DS authenticates access to the file share. This is a four-step process, as described in [Part one: enable AD DS authentication for your Azure file shares](#)

Components

- [Azure Storage](#) is a set of massively scalable and secure cloud services for data, apps, and workloads. It includes [Azure Files](#), [Azure Table Storage](#), and [Azure Queue Storage](#). In this

architecture, Azure Storage provides the underlying infrastructure for Azure Files. It hosts the cloud-based file shares that on-premises users access.

- [Azure Files](#) is a managed file storage service that provides file shares within an Azure Storage account. The files can be accessed from the cloud or on-premises. Windows, Linux, and macOS deployments can mount Azure file shares concurrently. File access uses the industry standard Server Message Block (SMB) protocol. In this architecture, Azure Files hosts the actual file shares that on-premises Windows Server environments securely access by using AD DS authentication.
- [Azure Virtual Network](#) is the fundamental building block for private networks in Azure. In this architecture, it provides the environment for Azure resources, such as virtual machines, to securely communicate with each other, with the internet, and with on-premises networks.
- [Azure ExpressRoute](#) is a service that extends on-premises networks into the Microsoft cloud through a private, dedicated connection. In this architecture, ExpressRoute ensures secure and reliable connectivity for accessing Azure-based file shares from on-premises systems.
- [Azure VPN Gateway](#) is a networking service that connects on-premises networks to Azure by using site-to-site VPNs, similar to connecting to a remote branch office. In this architecture, it provides an alternative to ExpressRoute for securely accessing Azure Files over Internet Protocol Security (IPsec) and Internet Key Exchange (IKE) protocols.
- [Azure Private Link](#) is a networking service that enables private connectivity from a virtual network to Azure platform as a service (PaaS), customer-owned, or Microsoft partner services. It simplifies this network architecture and secures the connection between endpoints in Azure by eliminating data exposure to the public internet.
- A private endpoint is a network interface that uses a private IP address from your virtual network. You can use private endpoints for your Azure Storage accounts to allow clients on a virtual network to access data over a private link.
- [Azure Firewall](#) is a managed, cloud-based network security service that protects your Azure Virtual Network resources. It's a fully stateful firewall service that has built-in high availability and unrestricted cloud scalability. In this architecture, you can configure Azure Firewall as a DNS proxy to simplify DNS management and eliminate the need for a dedicated virtual machine. A DNS proxy serves as an intermediary for DNS requests from client virtual machines to a DNS server.

Scenario details

Consider the following common scenario: an on-premises computer running Windows Server is used to provide file shares for users and applications. Active Directory Domain Services (AD DS) is used to help secure the files, and an on-premises DNS server manages network resources. Everything operates within the same private network.

Now assume that you need to extend file shares to the cloud.

The architecture described here demonstrates how Azure can meet this need cost-effectively while maintaining the use of your on-premises network, AD DS, and DNS.

In this setup, Azure Files is used to host the file shares. A site-to-site VPN or Azure ExpressRoute provides enhanced-security connections between the on-premises network and Azure Virtual Network. Users and applications access the files via these connections. Microsoft Entra ID and Azure DNS work together with on-premises AD DS and DNS to help ensure secure access.

In summary, if this scenario applies to you, you can provide cloud-based file shares to your on-premises users at a low cost while maintaining enhanced-security access via your existing AD DS and DNS infrastructure.

Potential use cases

- The file server moves to the cloud, but the users must remain on-premises.
- Applications that are migrated to the cloud need to access on-premises files, and also files that are migrated to the cloud.
- You need to reduce costs by moving file storage to the cloud.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

Reliability

Reliability helps ensure that your application can meet the commitments that you make to your customers. For more information, see [Design review checklist for Reliability](#).

- Azure Storage always stores multiple copies of your data in the same zone, so that it's protected from planned and unplanned outages. There are options for creating additional copies in other zones or regions. For more information, see [Azure Storage redundancy](#).
- Azure Firewall has built-in high availability. For more information, see [Azure Firewall Standard features](#).

Security

Security provides assurances against deliberate attacks and the misuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

These articles have security information for Azure components:

- [Azure security baseline for Azure Storage](#)
- [Azure security baseline for Azure Private Link](#)
- [Azure security baseline for Virtual Network](#)
- [Azure security baseline for Azure Firewall](#)

Cost Optimization

Cost Optimization focuses on ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

To estimate the cost of Azure products and configurations, use the Azure [Pricing calculator](#).

These articles have pricing information for Azure components:

- [Azure Files pricing](#)
- [Azure Private Link pricing](#)
- [Virtual Network pricing](#)
- [Azure Firewall pricing](#)

Performance Efficiency

Performance Efficiency refers to your workload's ability to scale to meet user demands efficiently. For more information, see [Design review checklist for Performance Efficiency](#).

- Your Azure Storage accounts contain all of your Azure Storage data objects, including file shares. A storage account provides a unique namespace for its data, a namespace that's accessible from anywhere in the world over HTTP or HTTPS. For this architecture, your storage account contains file shares that are provided by Azure Files. For best performance, we recommend the following:
 - Don't put databases, blobs, and so on, in storage accounts that contain file shares.
 - Have no more than one highly active file share per storage account. You can group file shares that are less active into the same storage account.
 - If your workload requires large amounts of IOPS, extremely fast data transfer speeds, or very low latency, then you should choose premium (FileStorage) storage accounts. A standard general-purpose v2 account is appropriate for most SMB file share workloads. For more information about the scalability and performance of file shares, see [Azure Files scalability and performance targets](#).

- Don't use a general-purpose v1 storage account, because it lacks important features. Instead, [upgrade to a general-purpose v2 storage account](#). The storage account types are described in [Storage account overview](#).
- Pay attention to size, speed, and other limitations. Refer to [Azure subscription and service limits, quotas, and constraints](#).
- There's little you can do to improve the performance of non-storage components, except to be sure that your deployment honors the limits, quotas, and constraints that are described in [Azure subscription and service limits, quotas, and constraints](#).
- For scalability information for Azure components, see [Azure subscription and service limits, quotas, and constraints](#).

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Rudnei Oliveira](#) | Senior Azure Security Engineer

Next steps

- [Quickstart: Create a virtual network using the Azure portal](#)
- [What is VPN Gateway?](#)
- [Tutorial: Create and manage a VPN gateway using Azure portal](#)
- [Azure enterprise cloud file share](#)
- [Azure Virtual Network concepts and best practices](#)
- [Planning for an Azure Files deployment](#)
- [Use private endpoints for Azure Storage](#)
- [Azure Private Endpoint DNS configuration](#)
- [Azure Firewall DNS settings](#)
- [Compare self-managed Active Directory Domain Services, Microsoft Entra ID, and managed Microsoft Entra Domain Services](#)

Related resources

- [Azure enterprise cloud file share](#)
- [Using Azure file shares in a hybrid environment](#)
- [Hybrid file services](#)

Enterprise file shares with disaster recovery

Azure NetApp Files

Microsoft Entra

Windows Server

This architecture provides file shares that fail over automatically to a backup region in case of failure. The failover is transparent to the clients and applications that access the shares. The shares can be used for applications and virtual desktops that must be resilient to disruption, whether planned or unplanned.

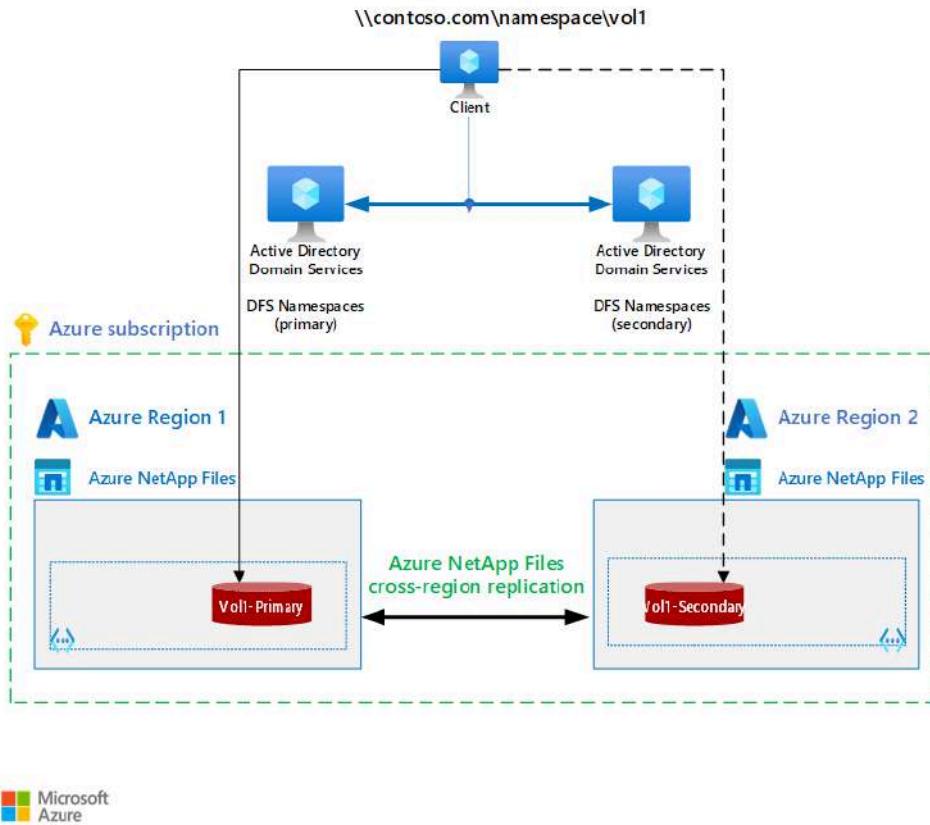
Azure NetApp Files provides the file shares. Its cross-region replication capability replicates the shares from the primary region to the secondary. Distributed File System (DFS) Namespaces in Windows Server can group shared folders on different servers into one or more logically structured namespaces.

Potential use cases

This architecture applies to businesses that want to provide file shares for clients or applications that must be resilient to unplanned outages or service maintenance events. Some examples are:

- Service Message Block (SMB) protocol file shares for desktop environments.
- SMB file shares for applications.

Architecture



Download a [Visio file](#) of this architecture.

- There are two Azure regions, a primary and a secondary.
- The Azure subscription includes a virtual network and an Azure NetApp Files account for each region.
- The cross-region replication feature of Azure NetApp Files replicates the files and folders from the primary region to the secondary region. This technique doesn't need virtual machines.
- Access to the file shares is managed by DFS Namespaces, a feature of Windows Server. You can think of it as Domain Name Server (DNS) for file shares.
- The Windows servers and Active Directory Domain servers can be hosted on Azure or on-premises.

Components

- [Azure NetApp Files](#) provides enterprise-grade Azure file shares that are powered by NetApp. Azure NetApp Files makes it easy for enterprises to migrate and run complex file-based applications with no code changes. It also provides a way to replicate data asynchronously from an Azure NetApp Files volume in one region to an Azure NetApp Files volume in another region. This capability provides data protection during region-wide outages or disasters. For more information, see [Cross-region replication of Azure NetApp Files volumes](#).

- DFS Namespaces is a role service in Windows Server that can group shared folders that are located on different servers into one or more logically structured namespaces. For more information, see [DFS Namespaces overview](#).

Alternatives

- Instead of Azure NetApp Files, you can use a Windows Server Scale-Out File Server cluster with custom replication of the file shares across regions. For more information, see [Scale-Out File Server for application data overview](#).
- Instead of Azure NetApp Files cross-region replication, you can use Azure File Sync to transform Windows Server into a quick cache of your Azure file shares. This might be appropriate for smaller file shares. For more information, see [Deploy Azure File Sync](#).

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

Reliability

Reliability helps ensure that your application can meet the commitments that you make to your customers. For more information, see [Design review checklist for Reliability](#).

Replicating to a second region increases availability by protecting against regional service interruptions.

- This solution has greater resiliency than a single-region deployment, and has failover capabilities.
- The secondary volume is read-only. It can be verified at any given time, increasing resiliency.
- You can run a disaster recovery test in isolation without interfering with the production deployment. The test uses the space-efficient volume clone feature to get a read/write copy of a volume in seconds.

Cost Optimization

Cost Optimization focuses on ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

The cost of the solution depends on the size of the volume that's replicated, the rate of change, and the destination tier of the Azure NetApp Files capacity pool. For more information, see [Azure NetApp Files pricing](#) or use the Azure [Pricing calculator](#).

See [Cost model for cross-region replication](#) for more examples.

Performance Efficiency

Performance Efficiency refers to your workload's ability to scale to meet user demands efficiently. For more information, see [Design review checklist for Performance Efficiency](#).

- Azure NetApp Files comes with three performance tiers: Standard, Premium, and Ultra. Cross-region replication can replicate between different tiers. When the primary region uses the Premium or Ultra tier, you can replicate to a lower tier, for example Standard. In case of a failover, you can then upgrade the tier of the secondary as required.
- The replication of the data is performed at the incremental block level—only changed data blocks are transferred—which minimizes data transfer.

This solution can be used for file shares ranging from 4 tebibytes (TiB) to a total volume of 12.5 pebibytes (PiB) on a single Azure NetApp Files account.

Deploy this scenario

To deploy on Azure, perform the following configuration tasks in the Windows Server DFS namespace:

1. Deploy the primary Azure NetApp Files account.
2. Create an SMB volume on the primary.
3. Deploy the secondary Azure NetApp Files account.
4. Replicate the volume to the secondary Azure NetApp Files account.
5. Configure DFS Namespaces to point to the primary volume.

In case of a failover:

1. Fail over the volumes of Azure NetApp Files.
2. Change the targets in DFS Namespaces.

These tasks can be and should be automated.

See [Disaster Recovery for Enterprise File Shares](#) for a step-by-step deployment guide.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal authors:

- [Max Melcher](#) | Cloud Solution Architect

Next steps

- [Register for NetApp Resource Provider](#)
- [Create a NetApp account](#)
- [Quickstart: Set up Azure NetApp Files and create an NFS volume](#)
- [Disaster Recovery for Enterprise File Shares](#)

Related resources

- [Hybrid file services](#)

Moodle deployment with Azure NetApp Files

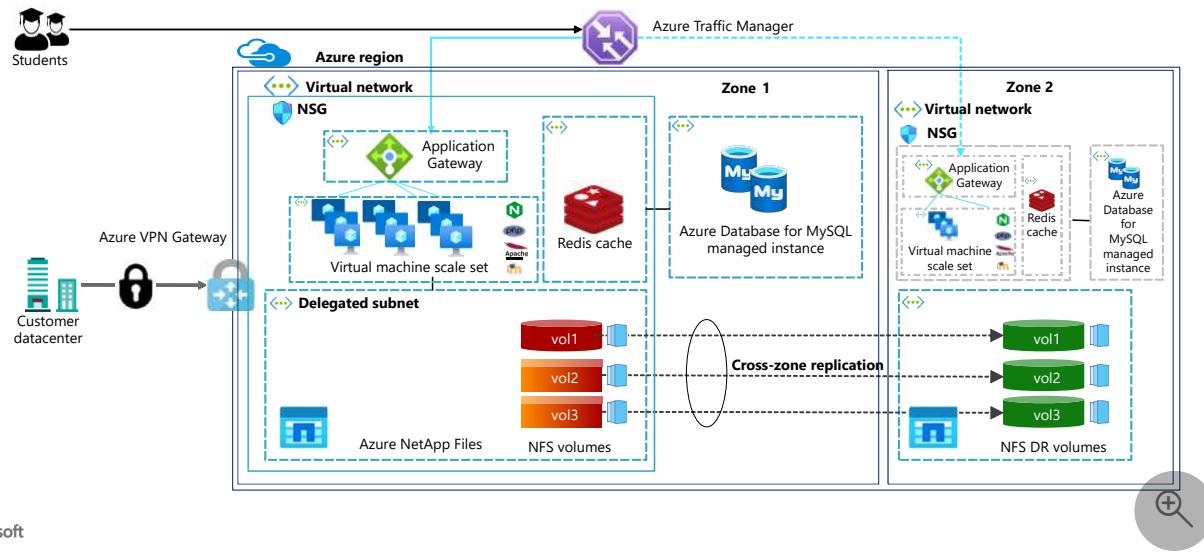
Azure Application Gateway Azure Managed Redis Azure Database for MySQL Azure NetApp Files
Azure Virtual Machine Scale Sets

Moodle is an open-source learning management system that requires high-throughput, low-latency access to storage. Many Moodle deployments require easy scalability to adapt to growing demand. This article explains how you can deploy Moodle by using Azure services on Azure Virtual Machine Scale Sets and store user-accessible learning data files in Azure NetApp Files. This article describes a zonal deployment for high availability and cross-zone replication and also gives examples of a single-zone deployment.

Architecture

For the best user experience, Moodle requires consistent low-latency access to scalable shared storage to meet the demands of office and home workers who use the service. Virtual Machine Scale Sets and Azure NetApp Files capacity pools and volumes can be sized up and down as the demand changes.

Highly available architecture for Moodle on Azure (multiple zones)



Download a [Visio file](#) of this architecture.

In addition to the Moodle deployment, the architecture uses Azure NetApp Files cross-zone replication to replicate the data volumes to a secondary zone. [Cross-zone replication](#) uses availability zones to provide high availability in a region and replication to a different zone in

the same region. A capacity pool that uses the Standard service level can host the destination data volumes during normal operation.

By using this approach, you don't need to start some components of the setup, like compute and ancillary services, during normal operation. As a result, you won't incur any operational cost for these components. You can also scale down the virtual machine scale sets to the minimum.

Only in a disaster recovery scenario should you start and scale up the necessary components to continue the service using the replicated data volumes. At this time, you can upgrade the service level of the destination Azure NetApp Files volumes to the Premium or Ultra service level if necessary.

After you recover the primary zone, the replication direction is reversed. The primary zone is updated with the changes that are applied during the failover, and the service can be failed back. Users are redirected to the failover zone through [Azure Traffic Manager](#), which operates at the DNS layer to quickly and efficiently direct incoming DNS requests based on the routing method of your choice.

Workflow

Students access the Moodle application data through an Azure Application Gateway where they can use Virtual Machine Scale Sets to build a scalable compute platform that runs the Moodle app to host users. Azure NetApp Files serves the content data to the Moodle app. Use a Redis cache for user session caching, locking, and key awareness. Store the learning content, student progress, and internal data in a MySQL database.

1. Insert learning content through a secure VPN gateway directly from the customer datacenter.
2. Students access the content through the application that's deployed on [Virtual Machine Scale Sets](#) through a secure application gateway.
3. You can scale the solution up or down depending on demand by adding or removing virtual machines (VMs) in the scale set and adjusting the [Azure NetApp Files volume service level](#).

Components

- [Moodle](#) is a free, open-source learning management system. In this architecture, Moodle serves as the core application that delivers educational content and tracks student progress.

- [Azure Database for MySQL Flexible Server](#) is a managed relational database service. In this architecture, it stores Moodle's structured data, including course content, user profiles, and student progress.
- [Azure Cache for Redis](#) is a secure, in-memory data store and messaging broker. In this architecture, it improves Moodle performance by caching user sessions, managing locks, and reducing load on the database.
- [Azure Virtual Machine Scale Sets](#) is an Azure compute service that you can use to deploy and manage a group of identical, load-balanced virtual machines. In this architecture, it hosts the Moodle application and automatically scales the number of VMs up or down based on demand.
- [Azure NetApp Files](#) is a high-performance file storage service. You can use this service to migrate and run the most demanding enterprise-file workloads in the cloud, such as native SMBv3, NFSv3, and NFSv4.1 file shares, databases, data warehouses, and high-performance computing applications. In this architecture, it stores Moodle's learning content and user-uploaded files. It provides scalable, low-latency access and cross-zone replication for high availability and disaster recovery.

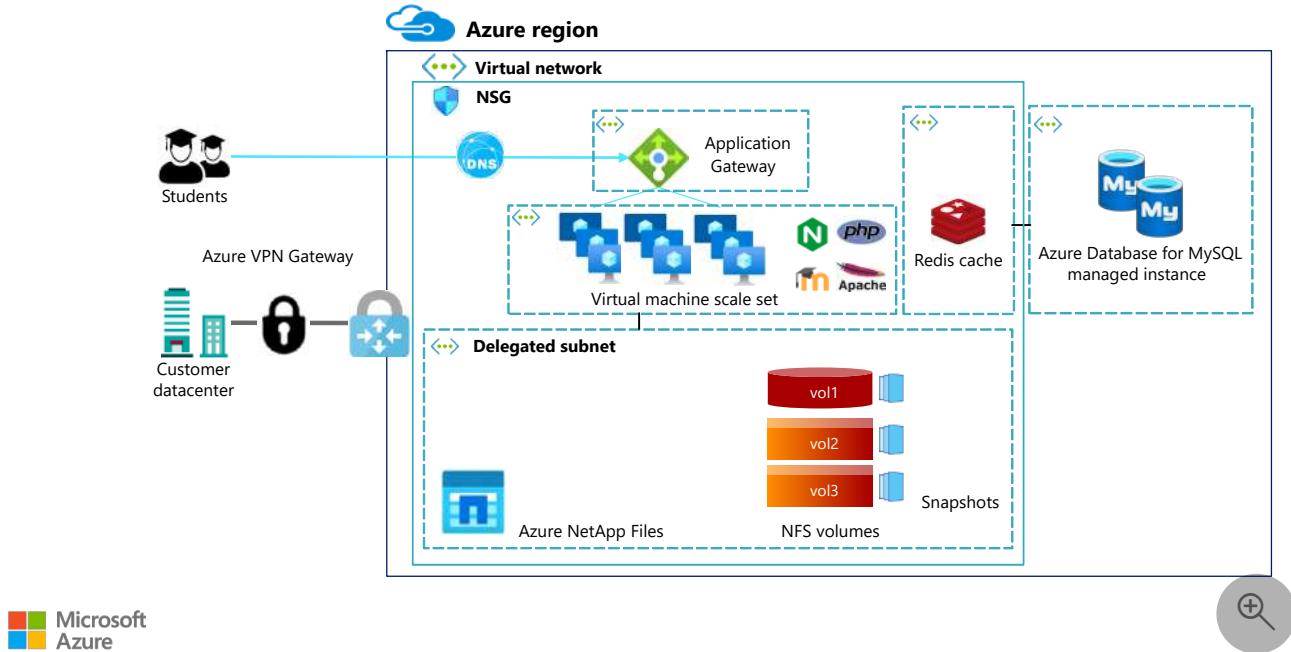
Alternatives

You can deploy the Moodle service by using any NFS-based shared file service that meets your requirements for low latency, high input or output operations per second, and throughput, especially for higher numbers of concurrent users. You can use an NFS service built on top of a set of Linux VMs, but this configuration can cause manageability, scalability, and performance challenges. Azure NetApp Files provides the lowest latency, best performance and scalability, and secure access to NFS shared storage.

Alternative deployments by using Azure NetApp Files

This diagram captures an example of a single-region deployment:

Reference architecture for Moodle on Azure (single region)



This single-region setup provides highly available access to the Moodle application and other components of the configuration.

Scenario details

This solution applies to Moodle deployments. Organizations that use Moodle span industries including education, business, IT, and finance.

This article outlines a solution that meets Moodle's needs. At the core of the solution is Azure NetApp Files, which is an Azure storage service. You can use this service to migrate and run the most demanding enterprise-scale file workloads in the cloud:

- Native Server Message Block (SMB) version 3, NFSv3, and NFSv4.1 file shares
- Database workloads
- Data warehouse workloads
- High-performance computing applications

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

Reliability

Reliability helps ensure that your application can meet the commitments that you make to your customers. For more information, see [Design review checklist for Reliability](#).

Azure NetApp Files is built on a bare-metal fleet of redundant, solid-state hardware. The service operates without interruption, even during maintenance operations. For more information about resiliency, see [Fault Tolerance, High Availability, and Resiliency in Azure NetApp Files](#).

Azure NetApp Files provides high availability for your stored data. For the Azure NetApp Files availability guarantee, see [SLA for Azure NetApp Files](#).

Security

Security provides assurances against deliberate attacks and the misuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

For all deployment options, you need to provide a valid Secure Shell (SSH) protocol 2 (SSH-2) RSA public–private key pair. The length should be at least 2,048 bits. Azure doesn't support other key formats such as ED25519 and ECDSA. Azure NetApp Files supports both customer-managed and platform-managed keys. These solutions provide unrestricted access to stored data, meet compliance requirements, and enhance data security. For more information and best practices for Azure NetApp Files security, see [Security FAQs for Azure NetApp Files](#).

Cost Optimization

Cost Optimization focuses on ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

For a medium-to-large-sized Moodle deployment of approximately 5,000 users with a 10% concurrency ratio, the recommended throughput is approximately 500 MB/s. This deployment can be built on a Linux-based Standard_D32s_v4 VM infrastructure that uses 8 TB of a P60-managed disk.

Azure NetApp Files provides a more cost-effective solution that uses 4 TiB of Ultra-service level capacity. For larger-scale applications that require more Azure NetApp Files capacity, both the Premium and Standard service levels provide sufficient performance. Use the Premium or Standard service level to improve cost effectiveness.

Use the [Azure pricing calculator](#) to estimate costs for Azure resources for your specific requirements. For more information, see [Azure NetApp Files cost model](#).

For a calculator that computes the Azure NetApp Files performance and total cost of ownership (TCO), see [Azure NetApp Files performance calculator](#). Use this calculator to find

the optimal balance between capacity, performance, and cost.

Operational Excellence

Operational Excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Design review checklist for Operational Excellence](#).

Performance Efficiency

Performance Efficiency refers to your workload's ability to scale to meet user demands efficiently. For more information, see [Design review checklist for Performance Efficiency](#).

You can scale this solution up or down as needed:

- Virtual Machine Scale Sets provides automatic scaling of resources. For more information, see [Overview of autoscale with Azure Virtual Machine Scale Sets](#).
- You can easily and nonintrusively scale the Azure NetApp Files capacity pools and volumes up and down to meet demand. For more information, see [Resize a capacity pool or a volume](#).
- You can adjust the Azure NetApp Files volume service level, which can be either Standard, Premium, or Ultra. The level that you choose affects the throughput limit of volumes with automatic quality of service. For more information, see [Performance considerations for Azure NetApp Files](#).

Deploy this scenario

For a deployment guide for Moodle on Azure NetApp Files, see [Azure NetApp Files for NFS storage with Moodle](#).

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Arnt de Gier](#) | Technical Marketing Engineer

Next steps

- [Moodle Cloud option ↗](#)
- [Azure Moodle directions on GitHub ↗](#)
- [Moodle docs: Redis cache store ↗](#)
- [Azure NetApp Files for NFS storage with Moodle ↗](#)
- [Solution architectures using Azure NetApp Files](#)
- [Automatic scaling with Virtual Machine Scale Sets flexible orchestration mode ↗](#)

Oracle Database with Azure NetApp Files

Azure NetApp Files

Azure Virtual Machines

Azure Virtual Network

The most demanding Oracle Database workloads require very high I/O capacity. They also need low-latency access to storage. This document describes a scalable, high-bandwidth, low-latency solution for running Oracle Database workloads on Azure virtual machines (VMs) with shared file access via the network file system (NFS) protocol. The architecture uses Azure NetApp Files, a first-party Azure shared file-storage service.

Benefits

Azure NetApp Files offers the following benefits:

- **Flexibility:** You can enlarge or reduce capacity and throughput on demand to align your configuration to the actual business needs without interruption to the service.
- **Scalability:** Use multiple storage volumes and add volumes on the fly to expand both capacity and throughput as needed
- **Availability:** Volumes are built on highly available fault-tolerant bare-metal fleet powered by ONTAP with built-in replication capabilities for business continuity and disaster recovery.
- **Consolidation:** Run multiple smaller database instances on an Azure VM while maintaining isolation of the database and log files over multiple storage volumes.
- **Data protection:** Space-efficient snapshot copies provide application-consistent point in time copies of live databases, and snapshot copies can be backed up by Azure NetApp Files backup or third-party solutions as desired.
- **Cloning:** Snapshots can be cloned to provide current data copies to test and development.
- **Storage throughput:** Networked storage is subjected to higher throughput limits than managed disk. As a result, you can use smaller VM SKUs than you would with managed disk storage without degrading performance. This approach could significantly reduce costs.

Potential use cases

This solution has many uses:

- Running new Oracle Database instances that require high availability (HA) and have high standards for performance.

- Migrating highly performant, highly available Oracle Database instances from on-premises to Azure Virtual Machines.
- Migrating Oracle Exadata systems to Azure.
- Consolidating multiple small Oracle instances onto a single Azure VM with one or more storage volumes for individual isolation and management.
- Cloning enterprise-scale Oracle Database systems for use in test and development environments. The solution is particularly suited for cases that require advanced data management capabilities. It can help meet aggressive data protection service level agreements (SLAs) by utilizing fast and space-efficient snapshots.
- Implementing Oracle Pacemaker clusters that use NFS shared storage.
- Deploying SAP AnyDB, or Oracle 19c.

Architecture

You can run a small-to-medium sized Oracle database on an Azure VM with one or more storage volumes for storing the database files, redo logs, and optionally a backup volume.



Azure region

Availability zone 1

VM subnet

Oracle VM



/oracle/db1-ora-data1
/oracle/db1-ora-log
/oracle/db1-ora-binary
/oracle/db1-ora-backup

10.0.1.1:/db1-ora-data1
10.0.1.2:/db1-ora-log
10.0.1.3:/db1-ora-binary
10.0.1.3:/db1-ora-backup

10.0.1.1



db1-ora-data1

10.0.1.2



db1-ora-log

10.0.1.3



db1-ora-binary
db1-ora-backup

Application volume group db1

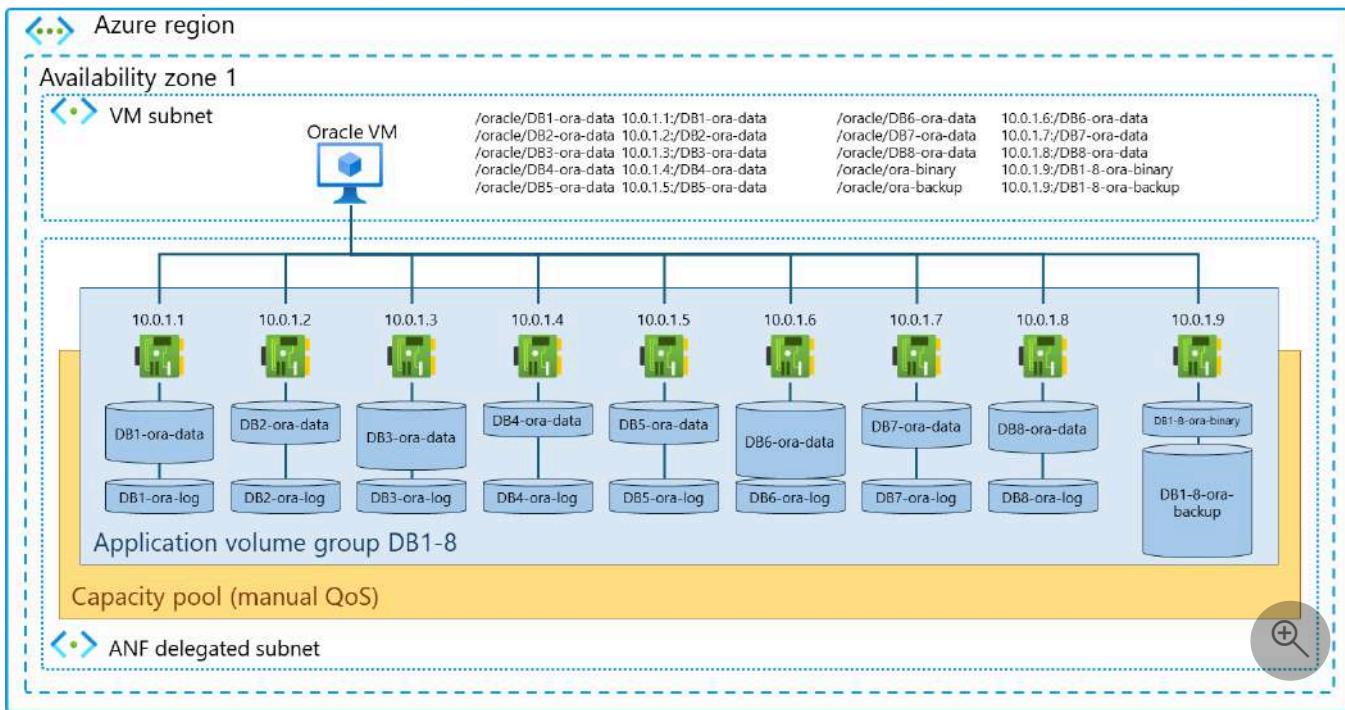
Capacity pool (manual QoS)



ANF delegated subnet



Deploy multiple data volumes for consolidating multiple smaller Oracle instances onto a single Azure VM.



Preparing the Azure NetApp Files service

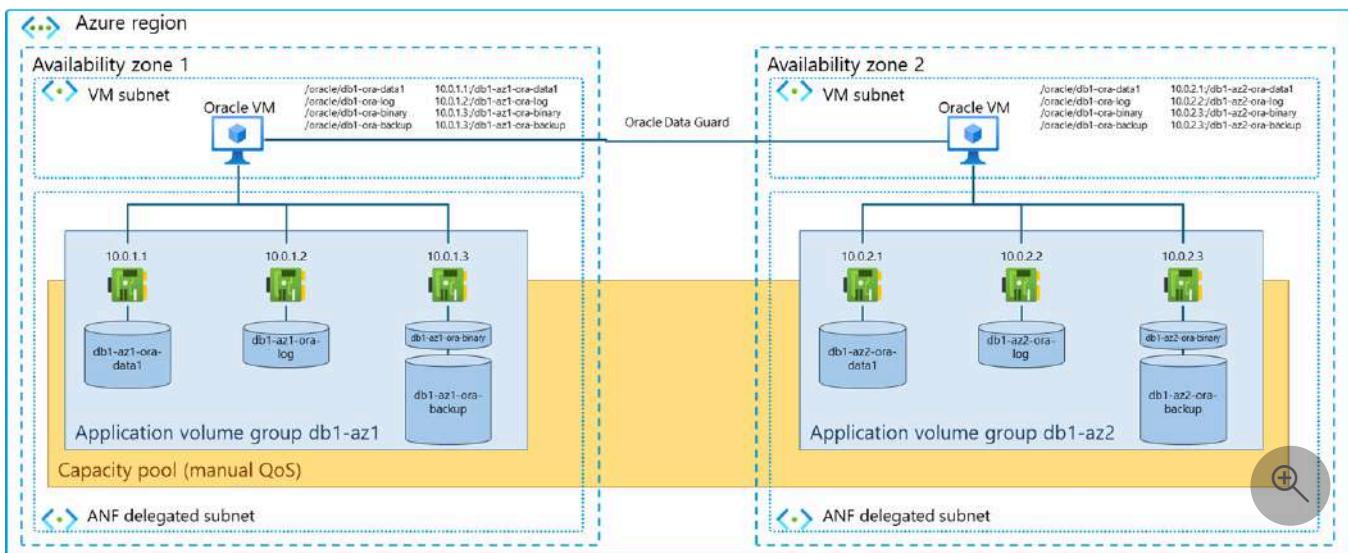
Create an Azure NetApp Files capacity pool of the desired capacity and service level. Check the [Quickstart for setting up Azure NetApp Files](#).

If you're migrating existing Oracle databases from on-premises to Azure, you can utilize AWR reports to obtain current throughput statistics which you need for sizing the Azure NetApp Files capacity pool and volumes. Recommendations for pool and volumes sizing can be obtained by processing [AWR reports through the Atroposs service](#). For more information about how to use the service, contact your Oracle on Azure specialist.

Available throughput for the volumes in a capacity pool is defined by the size and [service level \(Standard, Premium, or Ultra\)](#) of the selected capacity pool. Auto QoS capacity pools assign throughput to volumes directly related to the volume size. You can also assign throughput to volumes independently of their size, for which you can configure your capacity pool to use [manual QoS](#).

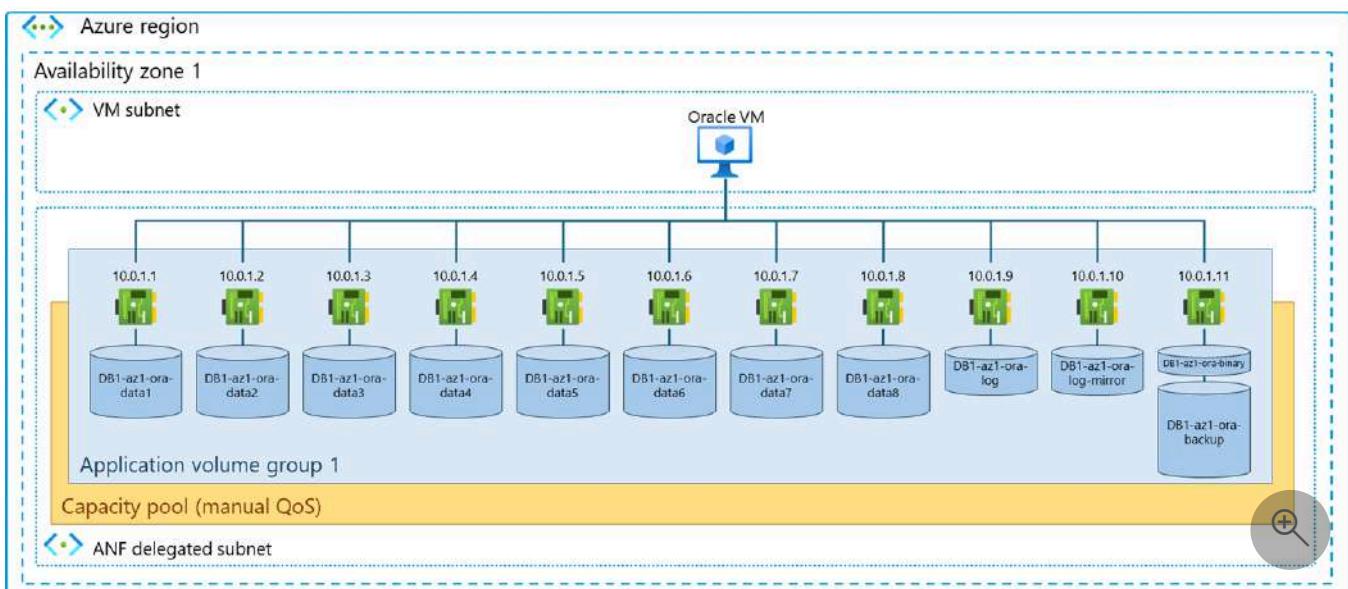
Data protection

To protect against unlikely zonal failures make use of Oracle Data Guard to replicate database files and redo logs to an alternate zone in the region.



Scalability

By using multiple storage volumes for database files, you can achieve additional scalability and flexibility. You can scale up to eight volumes for database files by using [application volume group for Oracle](#) to deploy the volumes. This approach helps place volumes in optimal locations within the Azure infrastructure for low-latency access by the VMs.



Components

The solution uses the following components:

- [Azure NetApp Files](#) is a first-party Azure file storage system that enables migrating and running file-based applications in Azure without code changes. It's developed by Microsoft and NetApp, a Microsoft partner.
- [Virtual Machines](#) is an infrastructure-as-a-service (IaaS) offer. You can use Virtual Machines to deploy on-demand, scalable computing resources. Virtual Machines provides

the flexibility of virtualization but eliminates the maintenance demands of physical hardware. This solution uses [Linux VMs with Oracle Database software](#).

- [Azure Virtual Network](#) is a networking service that manages virtual private networks in Azure. Through Virtual Network, Azure resources like VMs can securely communicate with each other, the internet, and on-premises networks. An Azure virtual network is like a traditional network operating in a datacenter. But an Azure virtual network also provides scalability, availability, isolation, and other benefits of the Azure infrastructure.
- [Oracle Database](#) is a multi-model database management system. It supports various data types and workloads.
 - The [dNFS](#) client optimizes I/O paths between Oracle and NFS servers. As a result, it provides better performance than traditional NFS clients.

Alternatives

This solution uses Oracle Data Guard (ODG) for disaster recovery (DR), and snapshots for local replication. A few options exist, as the following sections explain.

Cross-region replication

[Cross-region replication](#) provides efficient DR across regions in Azure. Cross-region replication uses storage-based replication. It doesn't use VM resources. For more information, see [Create volume replication for Azure NetApp Files](#).

Cross-zone replication

Cross-zone replication provides efficient HA across zones in Azure. Cross-zone replication uses the same highly efficient block-based replication with a minimum update interval of 10 minutes. This can be used to replicate the database files, while the redo log is replicated with Oracle Data Guard. For more information, see [Cross-zone replication of Azure NetApp Files volumes](#).

Availability sets and availability zones

ODG on Azure Virtual Machines functions like ODG in on-premises systems. But this product relies on its underlying architecture. If you run ODG on Azure VMs, consider also using one of these options to increase redundancy and availability:

- Place the Oracle VMs in the same availability set. This approach provides protection during these events:
 - Outages that equipment failures cause within a datacenter. VMs within an availability set don't share resources.

- Updates. VMs within an availability set undergo updates at different times.
- Place the Oracle VMs in different availability zones. This approach provides protection against the failure of an entire datacenter. Each zone represents a set of datacenters within a region. If you place resources in different availability zones, datacenter-level outages can't take all your VMs offline.

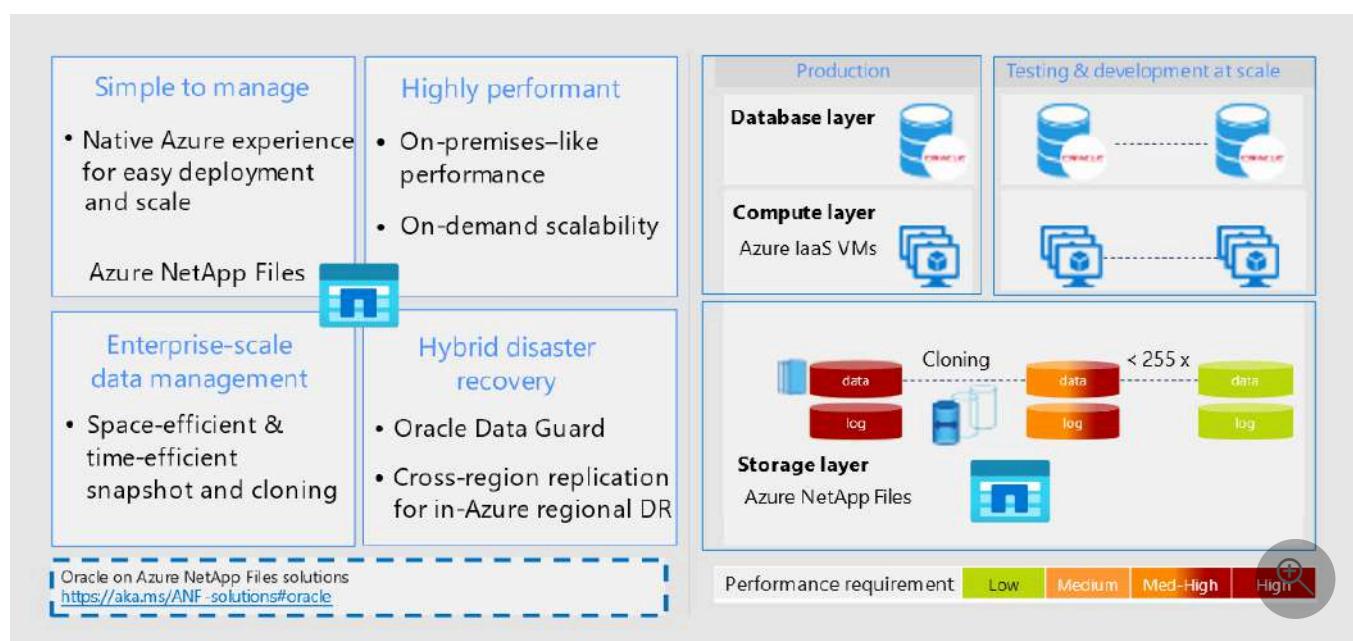
You can only choose one of these options. An Azure VM can't participate in availability sets and zones at the same time. Each option has advantages:

- Availability zones provide better availability than availability sets. See [SLA for Virtual Machines](#) for a comparison.
- You can place VMs that are in the same availability set in a [proximity placement group](#). This configuration minimizes the network latency between the VMs by guaranteeing that they're close to each other. In contrast, VMs that you place in different availability zones have greater network latency between them. It then takes longer to synchronize data between the primary and secondary replicas. As a result, the primary replica might experience delays. There's also an increased chance of data loss during unplanned failovers.

After you choose a solution, test it under load. Ensure that it meets SLAs for performance and availability.

Key benefits

This image shows the benefits of using Azure NetApp Files with Oracle Database.



Hosted service

As an Azure native service, Azure NetApp Files runs within the Azure datacenter environment. You can provision, consume, and scale Azure NetApp Files just like other Azure storage options. Azure NetApp Files uses reliability features that the NetApp data management software ONTAP provides. With this software, you can provision enterprise-grade NFS volumes for Oracle Database and other enterprise application workloads.

Low-latency performance

Azure NetApp Files uses a bare-metal fleet of all-flash storage. Besides using shared and highly scalable storage, Azure NetApp Files provides latencies of less than 1 millisecond. These factors make this service well-suited for using the NFS protocol to run Oracle Database workloads over networks.

The Azure DCsv2-series VMs can use high-performance, all-flash NetApp storage systems. These systems are also integrated into the Azure software-defined networking (SDN) and Azure Resource Manager frameworks. As a result, you get high-bandwidth, low-latency shared storage that's comparable to an on-premises solution. The performance of this architecture meets the requirements of the most demanding, business-critical enterprise workloads. For more information on the performance benefits of Azure NetApp Files, see [Benefits of using Azure NetApp Files with Oracle Database](#).

Azure NetApp Files offers on-demand scalability. You can enlarge or reduce deployments to optimize each workload's configuration.

Enterprise-scale data management

This solution can handle workloads that require advanced data management features. ONTAP provides functionality in this area that's unmatched in the industry:

- Space-efficient, instantaneous cloning enhances development and test environments.
- On-demand capacity and performance scaling makes efficient use of resources.
- Snapshots provide database consistency points and offer these benefits:
 - They're storage efficient. You only need limited capacity to create snapshots.
 - You can quickly create, replicate, restore, or clone them. As a result, they provide backup and recovery solutions that achieve aggressive recovery time objective (RTO) and recovery point objective (RPO) SLAs.
 - They don't affect volume performance.
 - They provide scalability. You can create them frequently and store many simultaneously.

Hybrid DR

The combination of ODG and Azure NetApp Files provides DR for this architecture. Those DR solutions are appropriate for cloud and hybrid systems. Their plans work across multiple regions and with on-premises datacenters.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

Reliability

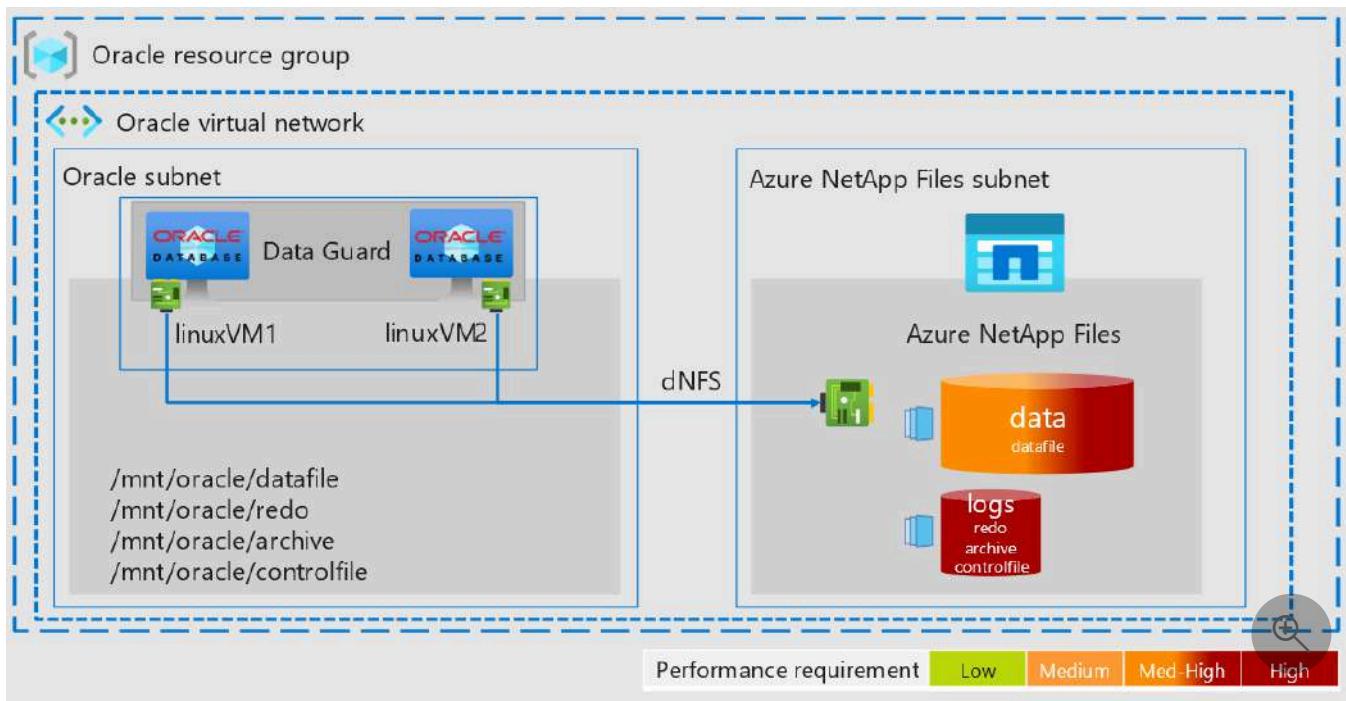
Reliability helps ensure that your application can meet the commitments that you make to your customers. For more information, see [Design review checklist for Reliability](#).

For Azure NetApp Files:

- See [SLA for Azure NetApp Files](#) for this service's availability guarantee.
- As [Enterprise-scale data management](#) discusses, you can use snapshots in backup and recovery solutions. Use Oracle hot backup mode and Azure NetApp Files APIs to orchestrate database-consistent snapshots.

When you use Oracle Database in Azure, implement a solution for HA and DR to avoid downtime:

- Use [ODG](#).
- Run the database on one virtual machine.
- Deploy a secondary VM, but only install the binaries on it.
- Put both VMs in the same virtual network. Then they can access each other over the private persistent IP address.



Security

Security provides assurances against deliberate attacks and the misuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

Azure NetApp Files secures data in many ways. For information about inherent protection, encryption, policy rules, role-based access control features, and activity logs, see [Security FAQs](#).

Cost Optimization

Cost Optimization focuses on ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

Using Azure NetApp Files instead of block storage can reduce costs:

- You can make the configuration cost-efficient. Traditional on-premises configurations are sized for maximum workload requirements. Consequently, these configurations are most cost-effective at maximum usage. In contrast, an Azure NetApp Files deployment is scalable. You can optimize the configuration for the current workload requirement to reduce expenses.
- You can use smaller VMs:
 - Azure NetApp Files provides low-latency storage access. With smaller VMs, you get the same performance that larger VMs deliver with ultra disk storage.
 - Cloud resources usually place limits on I/O operations. This practice prevents sudden slowdowns that resource exhaustion or unexpected outages can cause. As a result, VMs have disk throughput limitations and network bandwidth limitations. The network

limitations are typically higher than disk throughput limitations. With network-attached storage, only network bandwidth limits are relevant, and they only apply to data egress. In other words, VM-level disk I/O limits don't affect Azure NetApp Files. Because of these factors, network-attached storage can achieve better performance than disk I/O. This fact is true even when Azure NetApp Files runs on smaller VMs.

Smaller VMs offer these pricing advantages over larger ones:

- They cost less.
- They carry a lower Oracle Database license cost, especially when you use smaller, constrained-code SKUs.
- The network-attached storage doesn't have an I/O cost component.

These factors make Azure NetApp Files less costly than disk storage solutions.

Performance Efficiency

Performance Efficiency refers to your workload's ability to scale to meet user demands efficiently. For more information, see [Design review checklist for Performance Efficiency](#).

As the [Low-latency performance](#) section discusses, Azure NetApp Files provides built-in scalability.

Deploy this scenario

- For resources on deploying Oracle Database on Azure VMs with Azure NetApp Files, see [Solution architectures using Azure NetApp Files](#).
- For information on how to deploy and access Azure NetApp Files volumes, see [Azure NetApp Files documentation](#).
- Consider the database size:
 - For small databases, you can deploy all components, such as data files, the redo log, the archive log, and control files, into a single volume. Such simplified configurations are easy to manage.
 - For large databases, it's more efficient to configure multiple volumes. You can use [automatic or manual Quality of Service \(QoS\) volumes](#). These volume types provide more granular control over performance requirements.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Deanna Garcia](#) | Principal Program Manager
- [Arnt de Gier](#) | Technical Marketing Engineer for Azure NetApp Files

Next steps

- Oracle database performance on Azure NetApp Files single volumes
- Linux NFS mount options best practices for Azure NetApp Files
- Azure NetApp Files performance benchmarks for Linux
- Capacity management FAQs

Related resources

Fully deployable architectures that use Azure NetApp Files:

- [Run SAP BW/4HANA with Linux virtual machines on Azure](#)
- [Run SAP NetWeaver in Windows on Azure](#)

SQL Server on Azure Virtual Machines with Azure NetApp Files

Azure NetApp Files

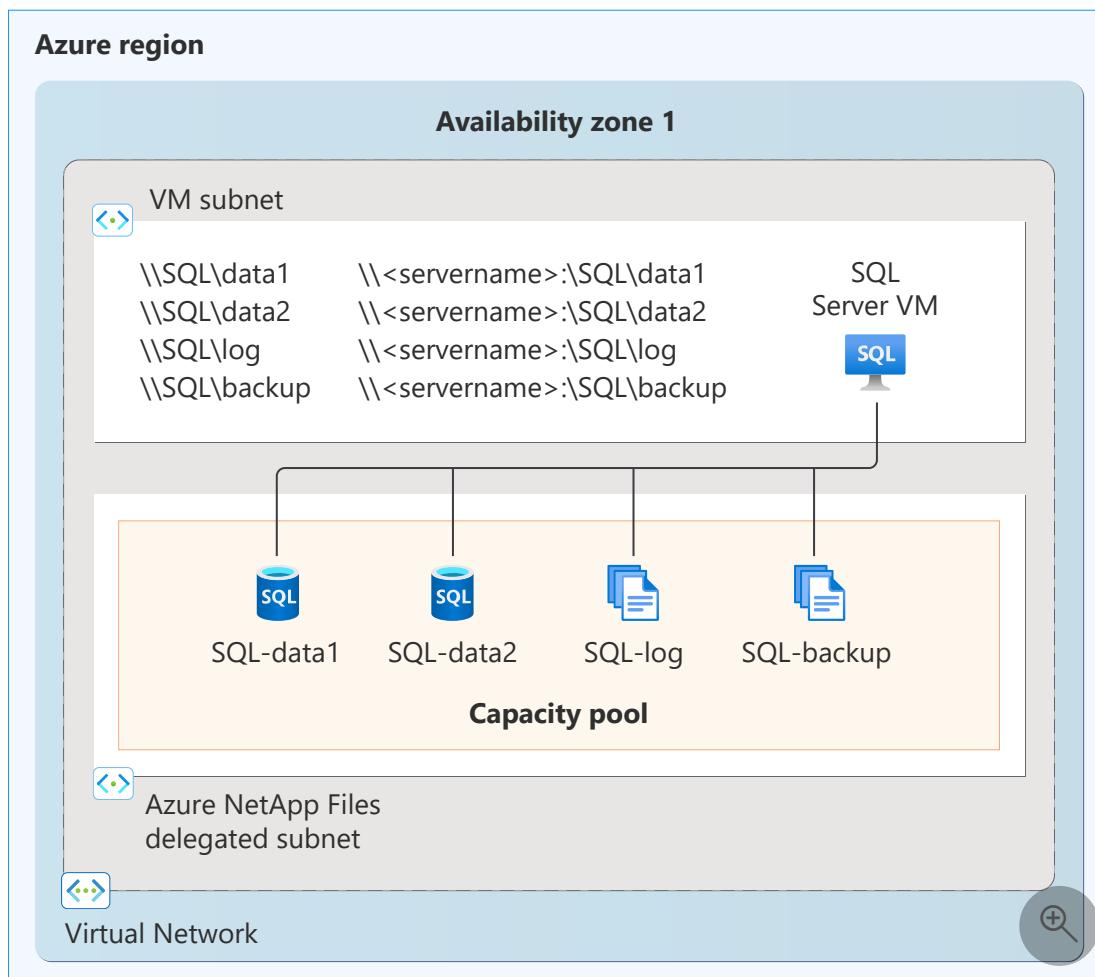
Azure SQL Server on Virtual Machines

Azure Virtual Machines

Azure Virtual Network

This article provides guidance about how to migrate SQL Server workloads to Azure Virtual Machines by using Azure NetApp Files. Azure NetApp Files is an enterprise-class file storage service that delivers high-performance, low-latency, and scalable storage via the Server Message Block (SMB) protocol. These capabilities make it well-suited for online transaction processing (OLTP) applications. This migration strategy enables cost-effective deployment while preserving enterprise-grade performance and availability.

Architecture



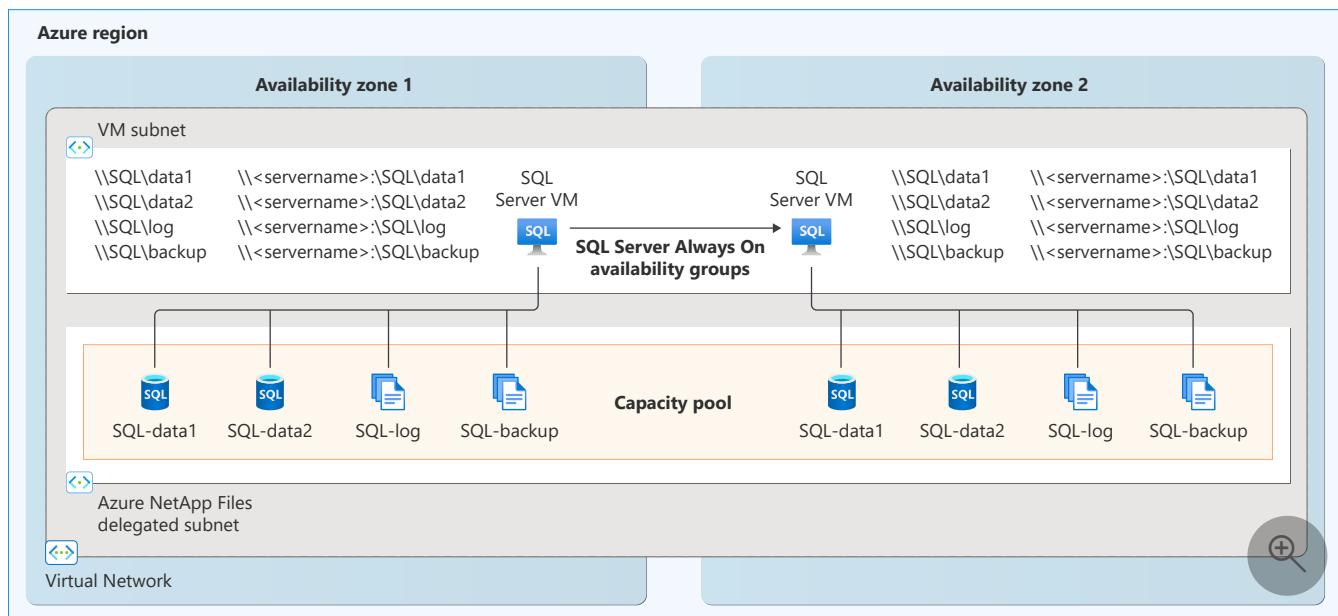
Workflow

Beginning with SQL Server 2012 (11.x), Database Engine user databases can be installed by using an SMB file server as the storage option. This capability also supports system databases (master, model, msdb, and tempdb), but their placement on SMB shares might be subject to

performance and configuration considerations. This option applies to both standalone SQL Server and SQL Server failover cluster instance (FCI) installations. For more information, see [Install SQL Server with SMB fileshare storage](#).

You can deploy SQL Server on Azure virtual machines (VMs) and use Azure NetApp Files to store the database and log files via SMB. We recommend that you enable [SMB continuous availability shares](#) for Azure NetApp Files to ensure SMB transparent failover. This failover allows for nondisruptive maintenance on the Azure NetApp Files service. You can [enable existing SMB volumes to use continuous availability](#).

You can also deploy a high availability workflow.



High availability and disaster recovery for SQL Server can be achieved on Azure by using [Always On failover clusters](#), with two databases on two separate VMs. Both VMs should be in the same virtual network to ensure that they can access each other over the private persistent IP address. You should place the VMs in the same [availability set](#) to allow Azure to place them into separate fault and upgrade domains. For geo-redundancy, set up the two databases to replicate between two different regions and configure [Always On availability groups](#).

Components

- An [Azure Windows-based VM](#) is a cloud-hosted infrastructure solution that provides the flexibility of virtualization and eliminates the maintenance demands of physical hardware. In this architecture, it serves as the compute layer.
- [Azure NetApp Files](#) is a high performance, Azure-native storage system. In this architecture, it stores database and log files.

- [SnapCenter](#) is a scalable platform that provides application-consistent data protection for both applications and databases. In this architecture, it's used for backup and recovery operations by creating space-efficient snapshots of the Azure NetApp Files volumes.

Alternatives

Other Azure solutions exist for providing storage for SQL Server on Azure VMs. When you evaluate alternatives, consider the benefits of space-efficient snapshots for primary data protection. These snapshots can be backed up to an Azure storage account by using [SnapCenter](#), which is included at no extra cost when used with Azure NetApp Files. This approach completes a comprehensive data protection and availability strategy.

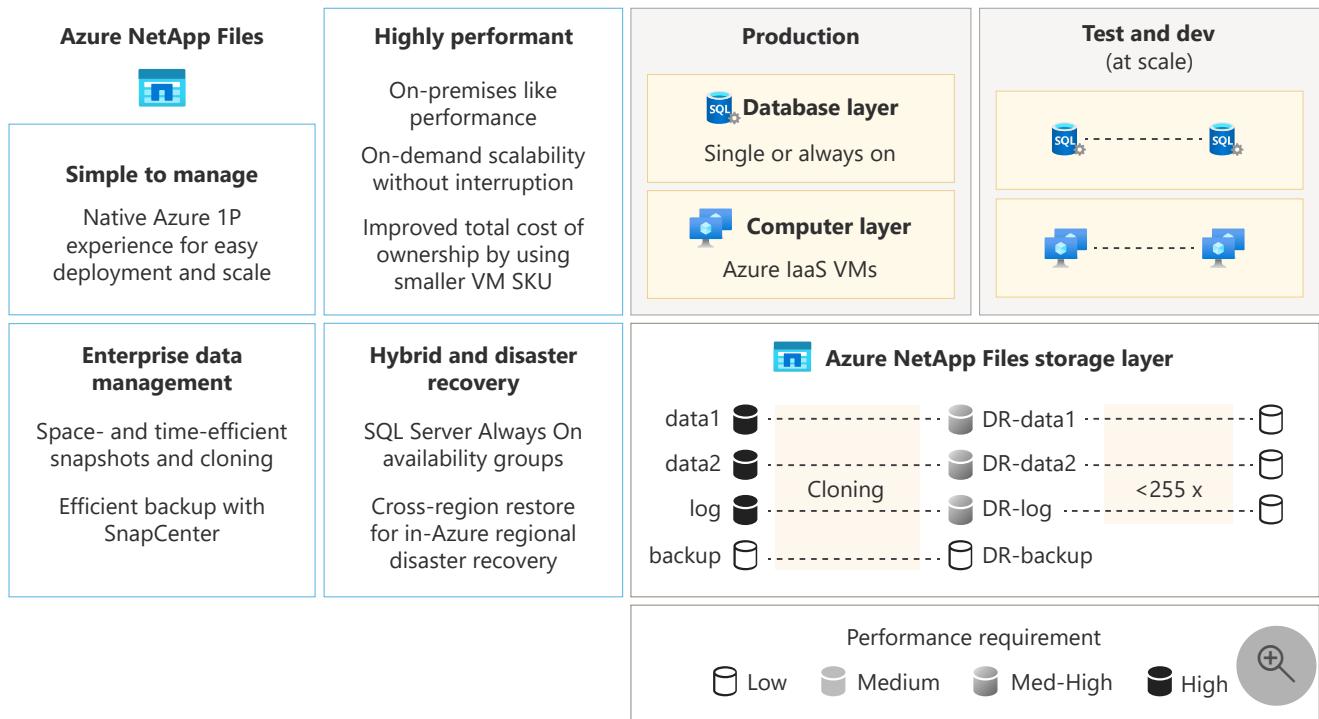
Scenario details

The most demanding SQL Server database workloads require high input/output (I/O) performance and low-latency access to storage. Azure provides low-latency, high-bandwidth shared file access via SMB through Azure NetApp Files. Azure VMs impose limits on I/O and bandwidth for managed disks. In contrast, Azure NetApp Files is subject only to egress network bandwidth limits. In other words, no VM-level storage I/O constraints apply to Azure NetApp Files.

Without I/O limits, SQL Server on smaller VMs that use Azure NetApp Files can perform as well or even better than SQL Server on larger VMs with disk storage. Azure NetApp Files provides flexibility that lets teams grow or shrink deployments on demand. This capability is unlike traditional on-premises configurations, which are sized for the maximum workload requirement and are most cost effective only at maximum utilization. In Azure with Azure NetApp Files, the configuration can be adjusted continually to the momentary workload requirement.

Azure NetApp Files is designed to meet the core requirements of running high-performance workloads like databases in the cloud. It provides the following advantages:

- Lower the total cost of ownership (TCO) compared to disk configurations
- Provide enterprise performance with low latency
- Maintain high availability
- Provide advanced data management



All components (database and log files) can initially be deployed into a single volume. This simplified configuration is easy to manage and well suited for smaller databases with low transaction activity.

For larger and more demanding databases, it's more efficient to configure multiple volumes and use a [manual quality of service \(QoS\) capacity pool](#), which allows for [more granular control over performance requirements](#).

Potential use cases

This solution applies to many use cases that include but aren't limited to the following scenarios:

- Migrate existing SQL Server instances that require high performance and high availability from on-premises to Azure on Azure VMs without rearchitecting.
 - Deploy SQL Server Always On failover cluster high availability architectures by using availability sets and [Azure NetApp Files continuous availability volume support](#).
 - Deploy enterprise-scale, hybrid or Azure-based disaster recovery architectures by using SQL Server Always On availability groups.
 - Enhance enterprise SQL Server environments that require advanced data management like fast cloning for test and development and stringent data protection service-level agreements (SLAs).

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

Reliability

Reliability helps ensure that your application can meet the commitments that you make to your customers. For more information, see [Design review checklist for Reliability](#).

High availability

Azure NetApp Files provides an SLA of 99.99% and is designed to handle hardware failures effectively. The 99.99% SLA can be improved for the highest levels of availability by using cross-zone replication in combination with [Always On availability groups](#).

Security

Security provides assurances against deliberate attacks and the misuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

Role-based access and data encryption

You can rely on [secure data plane concepts](#) with configurable role-based permissions at both the share and file levels. Data is encrypted in transit and at rest.

Cost Optimization

Cost Optimization focuses on ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

Smaller VM size

Network-attached storage can outperform disk-based storage because it relies on network bandwidth instead of disk I/O. Cloud resource constraints and the higher network limits of most VM SKUs contribute to this performance advantage. This solution supports smaller VM sizes with better performance. Smaller VMs are less costly and incur lower SQL Service license costs, while network-attached storage doesn't have an I/O cost factor.

Performance Efficiency

Performance Efficiency refers to your workload's ability to scale to meet user demands efficiently. For more information, see [Design review checklist for Performance Efficiency](#).

Scalability

Azure NetApp Files volumes can be expanded or contracted without interruption to the database. This flexibility supports both growth and cost reduction, without having to shut down and restart the database.

Deploy this scenario

For more information about how to deploy and access Azure NetApp Files volumes, see [Azure NetApp Files documentation](#).

Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal author:

- [Arnt de Gier](#) | Technical Marketing Engineer

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

- [Azure NetApp Files quickstart](#)
- [Create an SMB volume for Azure NetApp Files](#)
- [Benefits of using Azure NetApp Files for SQL Server deployment](#)
- [Windows Server failover cluster with SQL Server on Azure VMs](#)
- [Solution architectures that use Azure NetApp Files - SQL Server](#)

Hybrid file services

Microsoft Entra ID

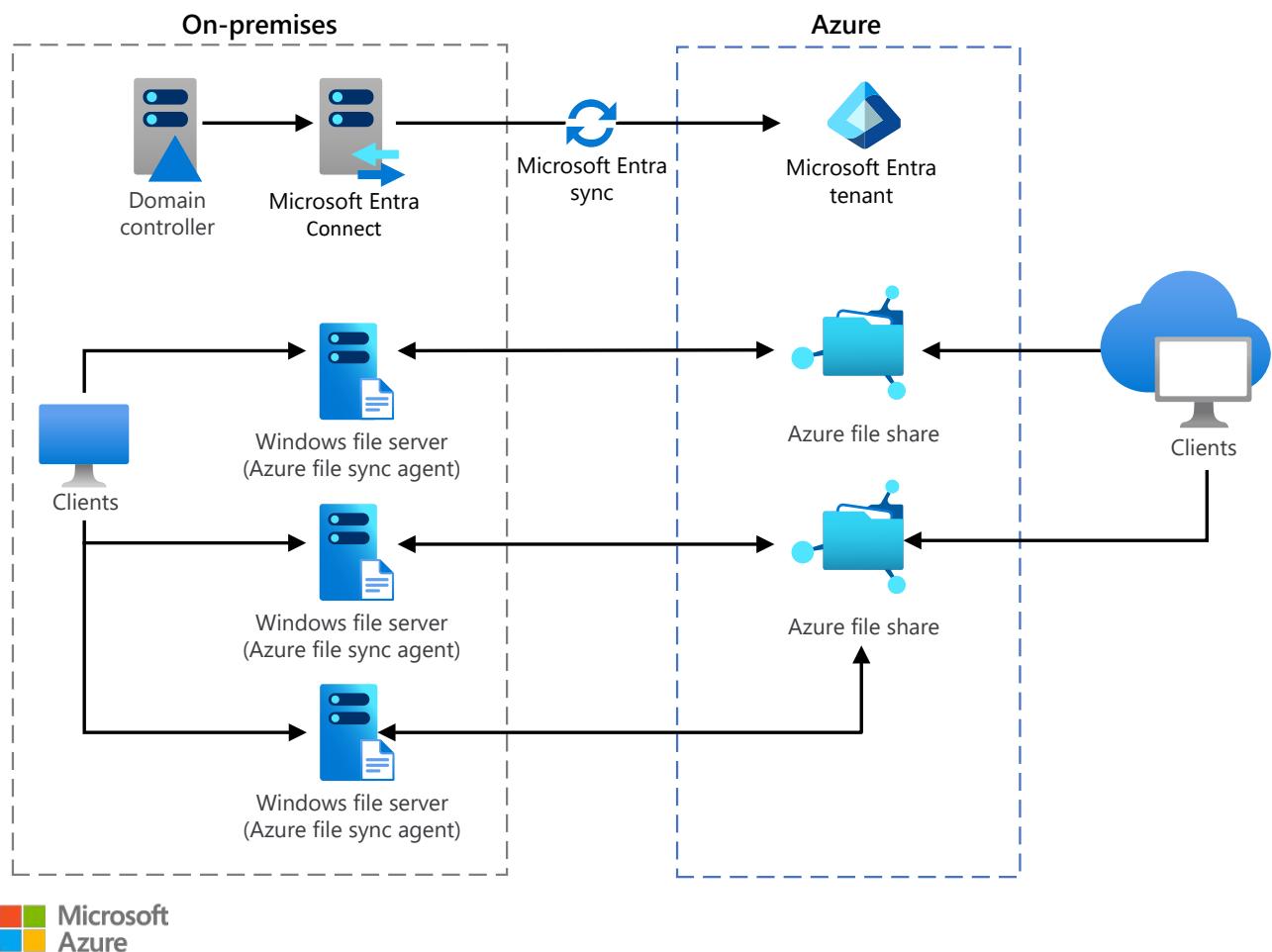
Azure ExpressRoute

Azure Files

Azure Storage Accounts

This reference architecture illustrates how to use Azure File Sync and Azure Files to extend file services hosting capabilities across cloud and on-premises file share resources.

Architecture



Download a [Visio file](#) of this architecture.

Workflow

The architecture consists of the following components:

- **Azure storage account**. A storage account that's used to host file shares.
- **Azure Files**. A serverless cloud file share that provides the cloud endpoint of a sync relationship by using Azure File Sync. Files in an Azure file share can be accessed directly with Server Message Block (SMB) or FileREST protocol.

- **Sync groups.** Logical groupings of Azure file shares and servers that run Windows Server. Sync groups are deployed into Storage Sync Service, which registers servers for use with Azure File Sync and contains the sync group relationships.
- **Azure File Sync agent.** This is installed on Windows Server machines to enable and configure sync with cloud endpoints.
- **Windows Servers.** On-premises or cloud-based Windows Server machines that host a file share that syncs with an Azure file share.
- **Microsoft Entra ID.** The Microsoft Entra tenant that's used for identity synchronization across Azure and on-premises environments.

Components

- [Azure Files](#) is a fully managed cloud file sharing service from Microsoft Azure that lets you create file shares accessible via SMB or network file system (NFS) protocols, with the scalability, security, and flexibility of the cloud. In this architecture, it enables scalable, secure file storage for applications and users and supports lift-and-shift scenarios and hybrid access.
- [Azure Storage accounts](#) are the foundational containers in Microsoft Azure that hold all your cloud storage data and provide a unified namespace for storing blobs, files, queues, tables, and disks. In this architecture, they serve as the foundational storage layer and ensure high availability, durability, and performance across workloads.
- [Microsoft Entra ID](#) is a cloud-based identity and access management service. In this architecture, it governs authentication, authorization, and conditional access for users, apps, and services across the environment.

Scenario details

Typical uses for this architecture include:

- Hosting file shares that need to be accessible from cloud and on-premises environments.
- Synchronizing data between multiple on-premises data stores with a single cloud-based source.

Recommendations

The following recommendations apply for most scenarios. Follow these recommendations unless you have a requirement that overrides them.

Azure Files usage and deployment

You store your files in the cloud in serverless Azure file shares. You can use them in two ways: by directly mounting them (SMB) or by caching them on-premises by using Azure File Sync. What you need to consider as you plan for your deployment depends on which of the two ways that you choose.

- Direct mount of an Azure file share. Because Azure Files provides SMB access, you can mount Azure file shares on-premises or in the cloud using the standard SMB client available in the Windows, macOS, and Linux operating systems. Azure file shares are serverless, so deploying them for production scenarios doesn't require managing a file server or network-attached storage (NAS) device. This means you don't have to apply software patches or swap out physical disks.
- Cache Azure file share on-premises with Azure File Sync. Azure File Sync makes it possible for you to centralize your organization's file shares in Azure Files, while keeping the flexibility, performance, and compatibility of an on-premises file server. Azure File Sync transforms an on-premises (or cloud) Windows Server into a quick cache of your Azure file share.

Deploy the Storage Sync Service

Begin Azure File Sync deployment by deploying a Storage Sync Service resource into a resource group of your selected subscription. We recommend provisioning as few Storage Sync Service objects as possible. You'll create a trust relationship between your servers and this resource. A server can only be registered to one Storage Sync Service. As a result, we recommend that you deploy as many Storage Sync Services as you need to separate groups of servers. Keep in mind that servers from different Storage Sync Services can't sync with each other.

Registering Windows Server machines with the Azure File Sync agent

To enable the sync capability on Windows Server, you must install the Azure File Sync downloadable agent. The Azure File Sync agent provides two main components:

- `FileSyncSvc.exe`. The background Windows service that's responsible for monitoring changes on the server endpoints and for initiating sync sessions.
- `StorageSync.sys`. A file system filter that enables cloud tiering and faster disaster recovery.

You can download the agent from the [Azure File Sync Agent Download](#) page at the Microsoft Download Center.

Operating system requirements

Azure File Sync is supported by the Windows Server versions that are listed in the following table.

[Expand table](#)

Version	Supported SKUs	Supported deployment options
Windows Server 2022	Azure, Datacenter, Essentials, Standard, and IoT	Full and Core
Windows Server 2019	Datacenter, Standard, and IoT	Full and Core
Windows Server 2016	Datacenter, Standard, and Storage Server	Full and Core
Windows Server 2012 R2	Datacenter, Standard, and Storage Server	Full and Core

For more information, see [Windows file server considerations](#).

Configuring sync groups and cloud endpoints

A *sync group* defines the sync topology for a set of files. Endpoints within a sync group are kept in sync with each other. A sync group must contain one *cloud endpoint*, which represents an Azure file share, and one or more server endpoints. A *server endpoint* represents a path on a registered server. A server can have server endpoints in multiple sync groups. You can create as many sync groups as you need to appropriately describe your desired sync topology.

A *cloud endpoint* is a pointer to an Azure file share. All server endpoints will sync with a cloud endpoint, making the cloud endpoint the hub. The storage account for the Azure file share must be located in the same region as the Storage Sync Service. The entirety of the Azure file share is synced, with one exception: a special folder, comparable to the hidden **System Volume Information** folder on an NT file system (NTFS) volume, is provisioned. This directory is called **.SystemShareInformation**, and it contains important sync metadata that doesn't sync to other endpoints.

Configuring server endpoints

A server endpoint represents a specific location on a registered server, such as a folder on a server volume. A server endpoint must be a path on a registered server (rather than a mounted share), and must use cloud tiering. The server endpoint path must be on a non-system volume. NAS isn't supported.

Azure file share to Windows file share relationships

You should deploy Azure file shares one-to-one with Windows file shares wherever possible. The server endpoint object gives you a great degree of flexibility on how you set up the sync topology on the server-side of the sync relationship. To simplify management, make the path of the server endpoint match the path of the Windows file share.

Use as few Storage Sync Services as possible. This simplifies management when you have sync groups that contain multiple server endpoints, because a Windows Server can only be registered to one Storage Sync Service at a time.

Pay attention to I/O operations per second (IOPS) limitations on a storage account when you deploy Azure file shares. The ideal is to map file shares one-to-one with storage accounts. It isn't always possible to do that because of various limits and restrictions from your organization and from Azure. When it's not possible to have only one file share deployed in a storage account, ensure that your most active file shares aren't in the same storage account.

Topology recommendations: firewalls, edge networks, and proxy connectivity

Consider the following recommendations for solution topology.

Firewall and traffic filtering

Based on the policies of your organization or on unique regulatory requirements, you might need to restrict communication with Azure. Therefore, Azure File Sync provides several mechanisms for configuring networking. Based on your requirements, you can:

- Tunnel the sync and file upload and download traffic over your Azure ExpressRoute or Azure virtual private network (VPN).
- Make use of Azure Files and Azure networking features such as service endpoints and private endpoints.
- Configure Azure File Sync to support your proxy in your environment.
- Throttle network activity from Azure File Sync.

To learn more about Azure File Sync and networking, see [Azure File Sync networking considerations](#).

Configuring proxy servers

Many organizations use a proxy server as an intermediary between resources inside their on-premises network and resources outside their network, such as in Azure. Proxy servers are useful for many applications, such as network isolation and security, and monitoring and logging. Azure File Sync can interoperate fully with a proxy server; however, you must manually configure the proxy endpoint settings for your environment with Azure File Sync. You do this by using the Azure File Sync server cmdlets in Azure PowerShell.

For more information on how to configure Azure File Sync with a proxy server, see [Azure File Sync proxy and firewall settings](#).

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

Reliability

Reliability helps ensure that your application can meet the commitments that you make to your customers. For more information, see [Design review checklist for Reliability](#).

- There are two main types of storage accounts for Azure Files deployments:
 - General purpose version 2 (GPv2) storage accounts. GPv2 storage accounts allow you to deploy Azure file shares on standard, hard disk-based (HDD-based) hardware. In addition to storing Azure file shares, GPv2 storage accounts can store other storage resources such as blob containers, queues, and tables.
 - FileStorage storage accounts: FileStorage storage accounts make it possible for you to deploy Azure file shares on premium, solid-state disk-based (SSD-based) hardware. FileStorage accounts can only be used to store Azure file shares. You can't deploy other storage resources such as blob containers, queues, and tables in a FileStorage account.
- You should ensure that Azure File Sync is supported in the regions where you deploy your solution. For more information, see [Azure File Sync region availability](#).
- You should ensure that the services that are referenced in the **Architecture** section are supported in the region where you deploy the hybrid file services architecture.
- To protect the data in your Azure file shares against data loss or corruption, all Azure file shares store multiple copies of each file as it's written. Depending on the requirements of your workload, you can select more degrees of redundancy.
- *Previous Versions* is a Windows feature that enables you to use server-side Volume Shadow Copy Service (VSS) snapshots of a volume to present restorable versions of a file

to an SMB client. VSS snapshots and Previous Versions work independently of Azure File Sync. However, cloud tiering must be set to a compatible mode. Many Azure File Sync server endpoints can exist on the same volume. You have to make the following PowerShell call per volume that has even one server endpoint, where you plan to or are using cloud tiering. For more information about Previous Versions and VSS, see [Self-service restore through Previous Versions and VSS \(Volume Shadow Copy Service\)](#).

Security

Security provides assurances against deliberate attacks and the misuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

- Azure File Sync works with your standard Active Directory Domain Services (AD DS) identity without any special setup beyond setting up Azure File Sync. When you use Azure File Sync, file access typically goes through the Azure File Sync caching servers rather than through the Azure file share. Because the server endpoints are located on Windows Server machines, the only requirement for identity integration is to use domain-joined Windows file servers to register with the Storage Sync Service. Azure File Sync stores access control lists (ACLs) for the files in the Azure file share, and replicates them to all server endpoints.
- Even though changes that are made directly to the Azure file share take longer to sync to the server endpoints in the sync group, you might want to ensure that you can enforce your AD DS permissions on your file share directly in the cloud also. To do this, you must domain join your storage account to your on-premises AD DS domain, just as your Windows file servers are domain joined. To learn more about domain joining your storage account to a customer-owned AD DS instance, see [Overview of Azure Files identity-based authentication options for SMB access](#).
- When you use Azure File Sync, there are three different layers of encryption to consider:
 - Encryption at rest for data that's stored in Windows Server. There are two strategies for encrypting data on Windows Server that work generally with Azure File Sync: encryption beneath the file system such that the file system and all of the data written to it is encrypted, and encryption within the file format itself. These methods can be used together if desired, because their purposes differ.
 - Encryption in transit between the Azure File Sync agent and Azure. Azure File Sync agent communicates with your Storage Sync Service and Azure file share by using the Azure File Sync REST protocol and the FileREST protocol, both of which always use HTTPS over port 443. Azure File Sync doesn't send unencrypted requests over HTTP.
 - Encryption at rest for data that's stored in the Azure file share. All data that's stored in Azure Files is encrypted at rest using Azure storage service encryption (SSE). Storage service encryption works much like BitLocker on Windows: data is encrypted beneath the file system level. Because data is encrypted beneath the file system of the Azure file

share as the data is encoded to disk, you don't need access to the underlying key on the client to read or write to the Azure file share.

Cost Optimization

Cost Optimization focuses on ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

- The [Azure Storage Pricing](#) page provides detailed pricing information based on account type, storage capacity, replication, and transactions.
- The [Data Transfers Pricing Details](#) article provides detailed pricing information for data egress.
- You can use the [Azure Storage Pricing Calculator](#) to help estimate your costs.

Operational Excellence

Operational Excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Design review checklist for Operational Excellence](#).

- The Azure File Sync agent is updated on a regular basis to add new functionality and to address issues. Microsoft recommends that you configure Microsoft Update to provide updates for the Azure File Sync agent as they become available. For more information, see [Azure File Sync agent update policy](#).
- Azure Storage offers soft delete for file shares so that you can recover your data when it's mistakenly deleted by an application or by another storage account user. To learn more about soft delete, see [Enable soft delete on Azure file shares](#).
- Cloud tiering is an optional feature of Azure File Sync that caches frequently accessed files locally on the server and tiers the others to Azure Files based on policy settings. When a file is tiered, the Azure File Sync file system filter (StorageSync.sys) replaces the file locally with a pointer to the file in Azure Files. A tiered file has both the **offline** attribute and the **FILE_ATTRIBUTE_RECALL_ON_DATA_ACCESS** attribute set in NTFS so that third-party applications can securely identify tiered files. For more information, see [Cloud Tiering Overview](#).

Performance Efficiency

Performance Efficiency refers to your workload's ability to scale to meet user demands efficiently. For more information, see [Design review checklist for Performance Efficiency](#).

- You should consider the type and performance of the storage account that you use to host Azure file shares. All storage resources that are deployed into a storage account share the limits that apply to that storage account. To find out more about determining the current limits for a storage account, see [Azure Files scalability and performance targets](#).

Next steps

- [What is Azure File Sync?](#)
- [How is Azure File Sync billed?](#)
- [How to plan for Azure File Sync Deployment?](#)
- [How to deploy Azure File Sync?](#)
- [Azure File Sync network considerations](#)
- [What is Cloud Tiering?](#)
- [What disaster recovery option are available in Azure File Sync?](#)
- [How to backup Azure File Sync?](#)

Related resources

Related hybrid guidance:

- [Hybrid architecture design](#)
- [Azure hybrid options](#)
- [Hybrid app design considerations](#)

Related architectures:

- [Azure enterprise cloud file share](#)
- [Azure Files accessed on-premises and secured by AD DS](#)
- [Use Azure file shares in a hybrid environment](#)

Mainframe file and tape backup to Azure using Luminex

Azure Event Hubs

Azure ExpressRoute

Azure SQL Database

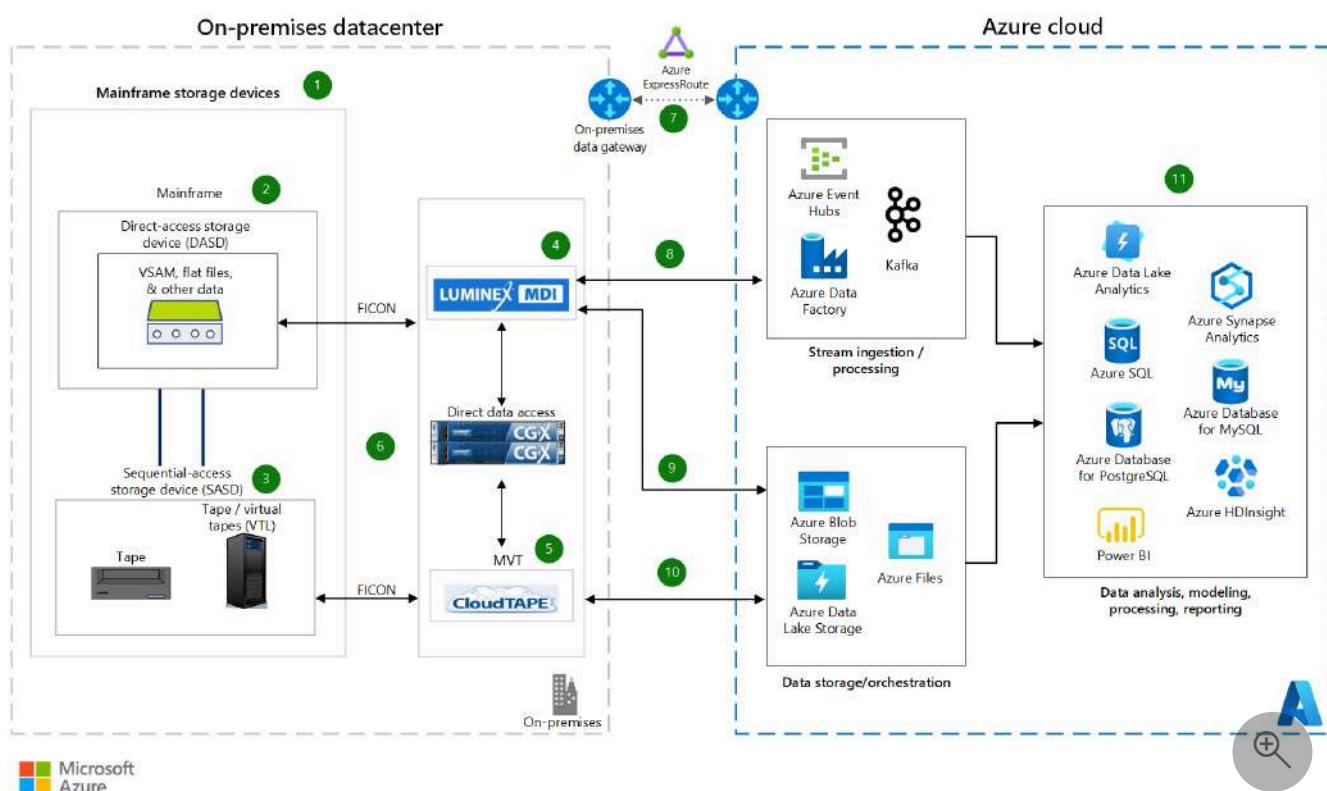
Azure Storage

Power BI

This article presents a solution for using Luminex products to transfer mainframe data to and from Azure to meet backup, archival, and other business needs. Key components in the solution include the Luminex mainframe data integration (MDI) platform's Cloud Data Sharing and the Luminex mainframe virtual tape (MVT) platform's CloudTAPE.

Apache® and Apache Kafka® are either registered trademarks or trademarks of the Apache Software Foundation in the United States and/or other countries. No endorsement by the Apache Software Foundation is implied by the use of these marks.

Architecture



Download a [Visio file](#) of this architecture.

Dataflow

1. On a mainframe, secondary storage devices include direct-access storage devices (DASDs) and sequential-access storage devices (SASDs).

2. DASDs are mounted on the mainframe.
3. A tape is a type of SASD that's attached to the mainframe as external storage.
4. The MDI platform sends information that can be stored on files to Azure. Examples include system management facilities (SMF) data, Virtual Storage Access Method (VSAM) files, sequential files, and generation data groups (GDGs). MDI hardware that's installed in the datacenter includes Luminex Channel Gateway X (CGX) controllers and Luminex MDI servers.
5. MVT CloudTAPE provides tape archival and backup. MVT hardware that's installed in the datacenter includes Luminex CGX controllers and CloudTAPE servers.
6. MDI and MVT use CGX controller devices that are based on the Fibre Connection (FICON) protocol. These devices connect directly to the mainframe. No System z Integrated Information Processor (zIIP) specialty engines are needed for data transfer. There's no Luminex agent on the mainframe, and no TCP/IP ports need to be open for communication between the mainframe and Luminex devices.
7. The mainframe data is transferred to Azure through a private, secure Azure ExpressRoute connection.
8. Luminex MDI zKonnect and other services stream the file data for big data analysis on Azure. For instance, system data like mainframe logs and SMF data is streamed to Azure Event Hubs. Azure services ingest the data and then process, transform, and project it.
9. MDI uses Luminex CGX devices to process, transfer, and cache file data. Two options are available:
 - Job control language (JCL) statements are submitted. Luminex provides statements that specify information about input files, the Azure destination, keys and security information, data transformation, and cloud file formats. Organizations that use the Luminex procedure for data transfer can use their own JCL statements. When the job finishes, a return code of zero indicates a successful transfer.
 - The job is monitored from the MDI UI. An operations team can use a combination of the scheduler, the mainframe, and the MDI UI to monitor and troubleshoot jobs. The MDI UI provides information like the job name, the job ID, the user or group, the start time, and the elapsed time. MDI retry mechanisms engage if the file transfer doesn't initially succeed.

The job can be configured to cache the files in local storage before the transfer. After the transfer finishes, that local storage is removed.

10. MVT CloudTAPE sends mainframe tape data to Azure data stores like Azure Blob Storage, Azure Files, and Azure Data Lake Storage. The data can be structured and unstructured. The transfer doesn't use JCL statements. Instead, MVT CloudTAPE moves or replicates mainframe tapes in IBM 3490 or 3590 format that CGX controllers emulate.

11. Azure services provide data processing, storage, analytics, and visualization capabilities.

Components

- [Azure Files](#) is a service that's part of [Azure Storage](#). Azure Files provides fully managed file shares in the cloud. Azure file shares are accessible via the industry-standard Server Message Block (SMB) protocol. In this architecture, Luminex MDI and MVT transfer mainframe files to Azure Files for cloud-based file storage and access.
- [Blob Storage](#) is a service that's part of Storage. Blob Storage provides optimized cloud object storage for unstructured data. In this architecture, Blob Storage provides a way to archive hot and cold mainframe data transferred from the mainframe systems.
- [ExpressRoute](#) is a connectivity service that extends on-premises networks to the Microsoft cloud by using a connectivity provider to establish private connections between on-premises data and Microsoft cloud services. In this architecture, ExpressRoute provides a private, secure connection to transfer mainframe data to Azure.
- In this solution, Luminex products can transfer mainframe data to several Azure databases:
 - [Azure SQL](#) is a family of Azure databases that are powered by the SQL Server engine. In this architecture, Azure SQL serves as a destination database for structured mainframe data transferred by Luminex MDI and MVT platforms.
 - [Azure SQL Database](#) is a platform as a service (PaaS) database engine that's part of the Azure SQL family. With AI-powered, automated features, SQL Database handles database management functions like upgrading, patching, backups, and monitoring. In this architecture, SQL Database serves as a managed database destination for structured mainframe data transferred by Luminex MDI and MVT platforms.
 - [Azure Database for PostgreSQL](#) is a managed relational database service that's based on the community edition of the open-source PostgreSQL database engine. In this architecture, Azure Database for PostgreSQL serves as an alternative managed database destination for structured mainframe data transferred by Luminex MDI and MVT platforms.

- [Azure Database for MySQL](#) is a managed relational database service that's based on the community edition of the open-source MySQL database engine. In this architecture, Azure Database for MySQL serves as another managed database destination option for structured mainframe data transferred by Luminex MDI and MVT platforms.
- [Event Hubs](#) is a managed big data streaming platform. In this architecture, Luminex zKonnect streams mainframe data to Event Hubs in near real time. It provides an endpoint that's compatible with Apache Kafka producer and consumer APIs for existing Apache Kafka client applications.
- [Power BI](#) is a collection of software services and apps that display analytics information. In this architecture, Power BI is used to turn mainframe data from various sources with varying structures into coherent, visually immersive, and interactive insights.
- [Data Lake Storage](#) is a scalable data storage service designed for big data analytics that provides low-cost, tiered storage and high throughput. In this architecture, Data Lake Storage serves as a destination for mainframe tape data transferred by MVT CloudTAPE, which enables big data analytics on mainframe datasets.

Alternatives

- Instead of using third-party solutions for data transfer, you can use a Microsoft solution. For information about transferring data from mainframe and midrange systems to Azure, see [Move archive data from mainframe systems to Azure](#). For information about specific Microsoft solutions, see the following resources:
 - [Copy files from a mainframe to Azure by using the Azure Data Factory FTP connector](#) ↗
 - [Transfer mainframe files to Azure by using SFTP](#) ↗
 - [Transfer a mainframe dataset to Azure Blob Storage by using a mainframe JCL](#) ↗
- To address any latency, connectivity, technological, and regulatory considerations, you can transfer data to Azure Stack instead of to Azure. Azure Stack Hub offers a set of cloud storage services. For more information, see [Azure Stack Hub storage: Differences and considerations](#).
- You can also use Luminex MVT and CGX devices for IBM z/VM and z/VSE mainframes.
- When you transfer tapes to Azure, you can compress and encrypt them to help transmit data safely at all stages. You can easily configure this functionality.
- You can also use this solution for bidirectional data interchange. You can recall the tape data to the mainframe and transform it into its original form.

- With MDI, the process is similar to the transfer to Azure. You submit JCL statements that provide the specifics of the reverse transfer. The data can be transferred as tapes or as sequential files. The JCL configuration specifies the format.
- With MVT CloudTAPE, the data is automatically recalled if you request it from the mainframe.
- Luminex CGX devices also support [Enterprise Systems Connection \(ESCON\)](#) ↗ channel connectivity. The existing mainframe backup software sees the channel gateway as a recognized mainframe tape device. As a result, no software change is needed.
- This solution uses ExpressRoute to transfer data from the datacenter to Azure. We recommend this approach, but you can also use the internet for data transfer.

Scenario details

Mainframe physical storage can be located on the mainframe processor, or it can be external to the mainframe. Processor storage, which is like memory for the mainframe, is located on the processor. Disk drives and tape drives are examples of external storage. Datasets in storage are organized into various logical record and block structures. Parameters like the dataset organization (DSORG) and record format (RECFM) define these data structures. Records in the dataset can be fixed or variable in length, and they can be stored in binary or text format.

Secondary storage devices like [DASDs](#) ↗ and [SASDs](#) ↗ store data that's either frequently or infrequently accessed.

- DASDs are used for immediate data location and retrieval. With direct access, you can read or write data by going directly to a specific physical location on the device. As a result, DASDs are fast and efficient.
- SASDs, such as tapes, are inherently slower than DASDs. To access tape data, you start at one location and then go through successive locations until you find the data that you need. Mainframes use physical tapes and [virtual tape libraries \(VTLs\)](#) ↗, which are also called *virtual tapes*. Currently, virtual tapes are preferred over physical tapes.

The type of storage that you use depends on your needs. Many organizations need cold storage for compliance, regulatory, reporting, audit, or other purposes. Some organizations have data retention policies that require you to store data for nearly 100 years. Examples of this type of data include copies of prescriptions, patient records, customer reward history, and other information. Data that you store for the long term is mostly high in volume and accessed infrequently. Long-term storage generally costs less than active storage, which you typically access multiple times a day and which is frequently updated. Security considerations also affect your choice of storage. Cyberattacks are a constant threat.

Azure offers various storage solutions and is a proven landing place for your storage, backup, and long-term archival needs. You can use cold storage for infrequently accessed data and hot storage for frequently accessed data. Mainframe file structures, such as VSAM datasets, flat files, and tape data, map to Azure data constructs within databases, structured files, and blob storage. Azure storage can store volume-intense data with cost efficiency, scalability, replication, and self-sustainability. Azure services can also help you retrieve your data, visualize your data, and gain insights from your data.

The solution in this article uses the [Luminex](#) MDI and MVT platforms to transfer mainframe data to and from Azure to meet backup, archival, and other business needs.

- [Luminex MDI](#) is a data transfer and coprocessing platform. MDI uses Luminex CGX devices to process, transfer, and cache mainframe files. MDI provides secure and efficient exchange of data and workload sharing between z/OS mainframes and distributed systems. By using MDI products like Cloud Data Sharing, Big Data Transfer, and zKonnect, you can move files to Azure for backup, archival, data normalization, merging, and analysis. You can configure the transferred data to arrive in ASCII or EBCDIC format in Azure. [MDI Cloud Data Sharing](#) provides a way to migrate mainframe files like VSAM files, sequential files, and GDGs to Azure. MDI also supports integration with Azure messaging services. Applications that are hosted on Azure can use the mainframe files that are stored on Azure for modernization, reduced latency, and improved performance.
- [Luminex MVT](#) is a tape archival and backup platform. MVT uses Luminex CGX control unit software that emulates mainframe 3490 and 3590 tape drives, so you can use existing tape applications without change. The CGX environment provides a suite of products for tape encryption, vaulting, migration, replication, retrieval, disaster recovery, and high availability. Specifically, the [CloudTAPE](#) product provides a way to migrate tape data to Azure.

MDI and MVT both use high-speed CGX controller devices to connect directly to the mainframe. These controllers are based on [FICON](#), a transport protocol that mainframe servers and attached enterprise-class storage controllers support. FICON uses Fibre Channel as the underlying transport protocol. The CGX controllers also take advantage of network attached storage (NAS) and internal storage systems to supply the high levels of performance, scalability, reliability, security, and availability that enterprises demand. With FICON transport, I/O can be shared across multiple systems. FICON delivers optimal protocol efficiency. It also helps provide data integrity and security, even with increased distances between server and storage devices.

With MDI and MVT, no zIIP specialty engines are needed for data transfer, and no TCP/IP ports need to be open to enable communication between the mainframe and Luminex devices. You plug the Luminex CGX devices directly into the mainframe just like any other mainframe

storage device. If necessary, your existing legacy backup and tape management software can run in parallel. For MVT CloudTAPE and MDI Cloud Data Sharing, the millions of instructions per second (MIPS) consumption is minimal because the transfer uses lightweight processes.

Potential use cases

Many scenarios can benefit from this solution. Possibilities include organizations with the following goals:

- Minimizing tape management and maintenance efforts.
- Modernizing legacy workloads.
- Finding backup and archival solutions.
- Extending their mainframe modernization by moving mainframe tapes to the cloud. Organizations might have this goal if they want to downsize their datacenter but not abandon it. If an organization doesn't use mainframe tapes heavily, the tapes might be a suitable candidate for migration.
- Transforming migrated data into a different format for cloud storage, such as converting EBCDIC data to ASCII, VSAM files to JSON, and sequential data to CSV format.
- Transferring tape metadata to Azure storage metadata.
- Providing new and refactored applications that are hosted on Azure with easy access to data.
- Expanding their cloud footprint.
- Easily monitoring, displaying, and reporting on mainframe files and tape data, and integrating this data with Azure services.
- Monetizing current and historical unlocked mainframe data and using it in cloud business intelligence and analytics tools.

If you're implementing a similar solution and want to share your experiences or feedback, contact the [Microsoft Legacy Modernization Azure Core Engineering \(ACE\) team](#).

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that can be used to improve the quality of a workload. For more information, see [Microsoft Azure Well-Architected Framework](#).

Reliability

Reliability ensures your application can meet the commitments you make to your customers. For more information, see [Design review checklist for Reliability](#).

- You can deploy this solution in multiple regions, and you can implement geo-replication in the data layer. Azure auto-failover groups also help provide data protection.
- Clustered CGX controllers can provide an active-active recovery solution during a failure.
- [MVT Synchronous Tape Matrix](#) provides reliability across multiple datacenters. Its infrastructure adjusts to failures without interruption.
- [Luminex Replication](#) can replicate data to one or many targets. A target can be one or more disaster recovery sites that each have a mainframe and CGX controller installed on the property. You can also preconfigure a target through Azure geo-replication. If you use Azure and other private or public clouds, you can also use a hybrid strategy for disaster recovery. Essentially, you can use the replication strategy that best meets your requirements. Examples include one-to-one, one-to-many, many-to-many, and cascading strategies.

Security

Security provides assurances against deliberate attacks and the abuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

- The fully managed storage in this solution eliminates issues that are related to physical media safety. Examples are damage or unauthorized access that might occur when you ship physical tapes in vehicles.
- [Luminex CGSafe](#) provides tape compression and encryption. This product is part of the MVT family and is included with CloudTAPE. CGSafe encrypts and compresses tapes during ingestion, at rest, and in transit.
- When you use MDI Cloud Data Sharing, files are sent over HTTPS by using SSL. In Azure, you can encrypt the files at rest.
- Because the solution uses FICON and ESCON connectivity, you don't need to open any ports for data transfer.

Cost Optimization

Cost Optimization is about looking at ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

- Pay-as-you-go pricing and multi-tiered models in Azure provide options to suit various cost and performance needs. For instance, if you access data infrequently, the Azure cool access tier is a good option for low-cost storage.
- The pricing of this solution depends on your volume of tape data, your datacenter location, and your bandwidth. The cost also depends on which Azure services you use.

These factors determine the hardware that you use, such as the number of Luminex CGX controllers. The factors also affect your software, service, licensing, and support costs.

- The data interchange doesn't require zIIP processors. As a result, you save on costs when you run the software.
- After the Luminex infrastructure is in place, you can use the Luminex hardware for other purposes. For instance, you might already use MDI Cloud Data Sharing for file transfer. If you augment your environment with MDI zKonnect for streaming, you can save on costs because you can purchase additional Luminex software and infrastructure at a significantly reduced price.
- If you already have an ExpressRoute infrastructure in place, you can use it for this solution.
- Using Azure and Luminex for backup and recovery helps you eliminate some costs that are associated with physical tape infrastructure. Examples include media and shipping expenses and off-site storage for vaulting.

Operational Excellence

Operational Excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Design review checklist for Operational Excellence](#).

- The data transfer to Azure in this solution gives you flexibility when you develop a backup strategy. You can enable automated, regular migration or phased-data migration. After you've installed a Luminex device in your datacenter, you can configure unidirectional or bidirectional communication, staged migration, or one-time migration. This flexibility provides support for implementing DevOps and Agile working principles and for immediate cloud adoption.
- You can take advantage of Azure capabilities for mainframe backup, archive, and disaster recovery.
- You can deploy continuous integration/continuous delivery (CI/CD) pipelines on Azure to manage data movement, transformation, and control activities.

Performance Efficiency

Performance Efficiency is the ability of your workload to scale to meet the demands placed on it by users in an efficient manner. For more information, see [Design review checklist for Performance Efficiency](#).

- If you have a high volume of data, you can cluster CGX controllers. Typically, one CGX device offers a data-transfer speed of up to 800 megabytes per second (MB/s). CGX controllers are available with up to four Fibre Channel ports or 1 Gigabit Ethernet (GbE),

10 GbE, or 25 GbE. These controllers also offer up to four ports for connectivity to attached storage systems.

- In Azure services, various performance options and tiers are available. For instance, block blob storage accounts offer standard and premium performance tiers. You can choose the tier that best meets your needs.
- Predefined access and life cycle management in Azure make it easy to optimize the performance of specific use cases.
- The tape emulation software in this solution uses the FICON I/O system. By using this system, you can reduce CPU time, increase data transmission speed, and reduce elapsed time.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal authors:

- [Daniel Saunders](#) | Sales Engineer
- [Bhuvi Vatsey](#) | Senior Technical Program Manager

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

- For more information, contact the [Microsoft Legacy Modernization Azure Core Engineering \(ACE\) team](#).
- For information about third-party data transfer solutions, see [Third-party archive solutions](#).

Related resources

- [Modernize mainframe and midrange data](#)
- [Move archive data from mainframe systems to Azure](#)
- [Replicate mainframe data by using Precisely Connect](#)
- [Mainframe and midrange data replication to Azure using Qlik](#)
- [Mainframe and midrange data replication to Azure using RDRS](#)
- [Migrate a mainframe data tier to Azure by using mLogica LIBER*IRIS](#)
- [Mainframe modernization using BMC AMI Cloud](#)

Mainframe file replication and sync on Azure

Azure Data Factory

Azure Data Lake

Azure SQL Database

Azure Storage

Azure Virtual Machines

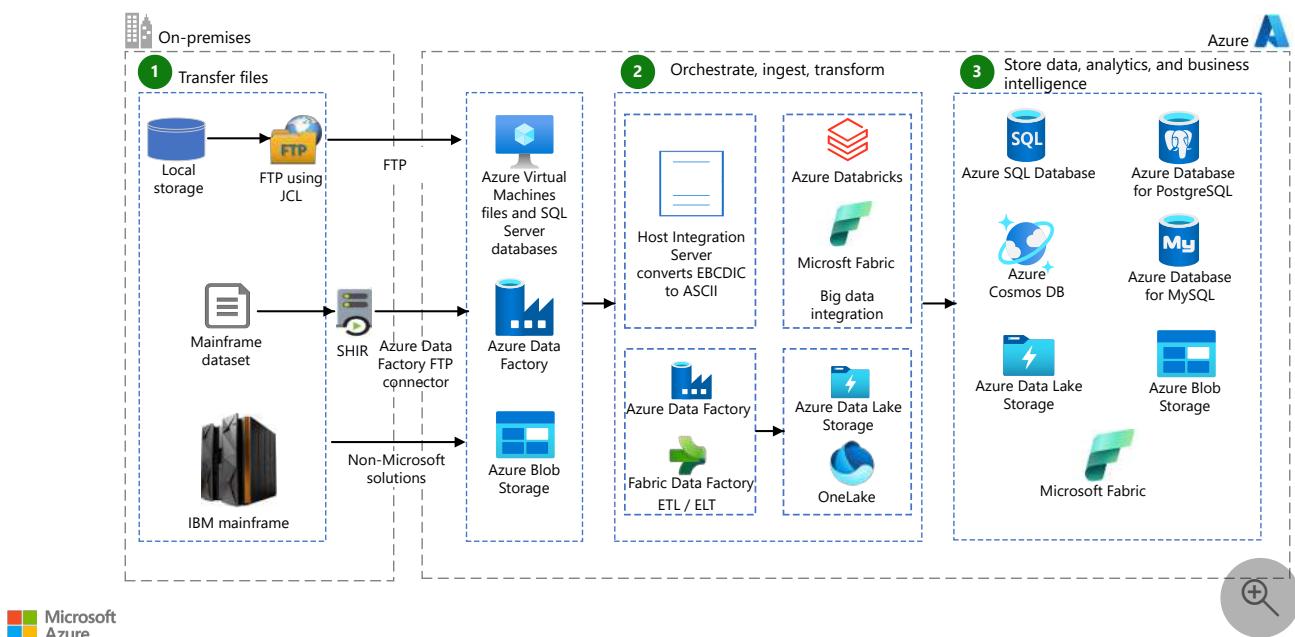
Solution ideas

This article describes a solution idea. Your cloud architect can use this guidance to help visualize the major components for a typical implementation of this architecture. Use this article as a starting point to design a well-architected solution that aligns with your workload's specific requirements.

When you migrate an on-premises mainframe or midrange application to Azure, data transfer is a key consideration. Several modernization scenarios require you to replicate files to Azure quickly or to maintain synchronization between on-premises files and Azure files.

This article describes several ways to transfer files to Azure, convert and transform file data, and store the data on-premises and in Azure.

Architecture



Download a [Visio file](#) of this architecture.

Dataflow

The following dataflow corresponds to the architecture diagram:

1. Transfer files to Azure:

- The easiest way to transfer files on-premises to Azure is by using [File Transfer Protocol \(FTP\)](#). You can host an FTP server on an Azure virtual machine (VM). A simple FTP job control language (JCL) sends files to Azure in binary format, which is essential to preserving mainframe and midrange computation and binary data types. You can store transmitted files in on-premises disks, Azure VM file storage, or Azure Blob Storage.
- You can also upload on-premises files to Blob Storage by using tools like [AzCopy](#).
- The Azure Data Factory FTP or Secure File Transfer Protocol (SFTP) connector can be used to transfer data from the mainframe system to Blob Storage. This method requires an intermediate VM on which a self-hosted integration runtime is installed.
- You can also find non-Microsoft tools on [Azure Marketplace](#) to transfer files from mainframes to Azure.

2. Orchestrate, convert, and transform data:

- Azure can't read IBM Extended Binary Coded Decimal Interchange Code (EBCDIC) code page files in Azure VM disks or Blob Storage. To make these files compatible with Azure, Host Integration Server (HIS) converts them from EBCDIC to American Standard Code for Information Interchange (ASCII) format.

Copybooks define the data structure of COBOL, PL/I, and assembly language files. HIS converts these files to ASCII based on the copybook layouts.

- Mainframe file data conversion can be achieved by using the Azure Logic Apps connector for IBM host files.
- Before you transfer data to Azure data stores, you might need to transform the data or use it for analytics. Azure Data Factory can manage these extract-transform-load (ETL) and extract-load-transform (ELT) activities and store the data directly in Azure Data Lake Storage. Alternatively, you can use Fabric Data Factory and OneLake store.
- For big data integrations, Azure Databricks, as well as Microsoft Fabric, can perform all transformation activities fast and effectively by using the Apache Spark engine for in-memory computations.

3. Store data:

You can store transferred data in one of several available persistent Azure storage modes, depending on your requirements.

- If analytics aren't required, Azure Data Factory can store data directly in a wide range of storage options, such as Data Lake Storage, Blob Storage, and Microsoft Fabric OneLake.
- [Azure hosts various databases](#) that address different needs:
 - Relational databases include the SQL Server family and open-source databases like PostgreSQL and MySQL.
 - Nonrelational databases include Azure Cosmos DB, which is a fast, multi-model, globally distributed NoSQL database.

Review analytics and business intelligence. [Microsoft Fabric](#) is an all-in-one analytics solution that covers everything from data movement to data science, real-time analytics, and business intelligence. It offers a suite of services, including data lake, data engineering, and data integration, all in one place.

Components

This architecture uses the following components.

Networking

An [on-premises data gateway](#) is bridge software that connects on-premises data sources to cloud services. In this architecture, it enables communication between mainframe systems and Azure services for file transfer and integration. You can install the gateway [on a dedicated on-premises VM](#).

Data integration and transformation

This architecture outlines various Azure-native migration tools that you can use based on your mainframe source data and target database.

- [Data Provider for Host Files](#) is a component of [HIS](#) that converts EBCDIC code page files to ASCII. The provider can read and write records offline in a local binary file. Or it can use Systems Network Architecture (SNA) or Transmission Control Protocol/Internet Protocol (TCP/IP) to read and write records in remote IBM z/OS mainframe datasets or i5/OS physical files. HIS connectors are available for [BizTalk](#) and [Logic Apps](#). In this architecture, Data Provider for Host Files enables file-level access and transformation of IBM z/OS and i5/OS datasets for migration to Azure.

- [Azure Data Factory](#) is a hybrid data integration service that you can use to create, schedule, and orchestrate ETL and ELT workflows. In this architecture, Azure Data Factory transfers mainframe files to Blob Storage via FTP and manages transformation pipelines.
- [Azure Databricks](#) is an Apache Spark-based analytics platform optimized for Azure. In this architecture, it enriches and correlates incoming mainframe data with other datasets for advanced analytics and transformation.
- [Microsoft Fabric](#) is an intelligent data platform with a suite of cloud services and tools for every data life cycle stage, including ingestion, preparation, storage, analysis, and visualization. In this architecture, Fabric enables organizations to study data movement, experiment with data science, and perform real-time analytics and business intelligence on transformed mainframe data.
- [Logic Apps](#) is a cloud-based service that you can use to automate workflows and integrate applications, data, and services across different environments. In this architecture, it uses the IBM Host File connector to interact with mainframe systems and automate file parsing and transformation.

Databases

This architecture outlines the process of migrating mainframe file data to cloud storage and managed databases in Azure. It includes converting mainframe file metadata to match the target schema in Azure.

- [Azure SQL Database](#) is a scalable relational cloud database service. SQL Database is evergreen and always up-to-date, with AI-powered and automated features that optimize performance and durability. Serverless compute and hyperscale storage options automatically scale resources on demand. In this architecture, SQL Database stores transformed mainframe data and supports high availability. It also supports cost efficiency through [Azure Hybrid Benefit](#) because you can use your existing on-premises SQL Server licenses on the cloud with no extra cost.
- [Azure SQL Managed Instance](#) is a platform as a service (PaaS) offering that provides full SQL Server compatibility with managed infrastructure. In this architecture, it modernizes legacy applications by hosting migrated mainframe data with minimal code changes.
- [SQL Server on Azure Virtual Machines](#) is an infrastructure as a service (IaaS) solution that lifts and shifts SQL Server workloads to Azure, which combines the flexibility and hybrid connectivity of Azure with SQL Server performance, security, and analytics. In this architecture, it provides control over SQL Server configurations for hosting mainframe-derived data.

- [Azure Database for PostgreSQL](#) is a managed open-source relational database service. In this architecture, it serves as a target for migrated mainframe data that requires PostgreSQL compatibility.
- [Azure Database for MySQL](#) is a managed MySQL database service. In this architecture, it supports workloads that require MySQL-based storage for transformed mainframe data.
- [Azure Cosmos DB](#) is a globally distributed NoSQL database service that includes multi-model support. In this architecture, it stores high-performance, scalable applications built on transformed mainframe data.

Other data stores

- [Blob Storage](#) is a cloud-based object storage solution that stores large amounts of unstructured data, such as text or binary data. You can access this data from anywhere via HTTP or HTTPS. You can use Blob Storage to expose data publicly or to store application data privately. In this architecture, it stores binary and text files transferred from mainframe systems and serves as a staging area for transformation.
- [Data Lake Storage](#) is a storage repository that holds a large amount of data in native, raw format. Data Lake Storage provides scaling for big data analytics workloads with terabytes and petabytes of data. The data typically comes from multiple heterogeneous sources, and can be structured, semi-structured, or unstructured. In this architecture, it stores raw and transformed mainframe data in native format for processing by analytics services.
- [OneLake in Microsoft Fabric](#) is a single, unified, logical data lake. In this architecture, it serves as the storage destination for Fabric Data Factory pipelines. It provides a centralized location to store transformed mainframe data for analytics and business intelligence workloads.

Scenario details

Converting mainframe files from EBCDIC-encoded format to ASCII format is necessary for migrating data from mainframe systems to Azure cloud storage and databases. Mainframe applications generate and handle large amounts of data daily. This data must be accurately converted for use in other platforms.

As your organization transitions mainframe file system data, you should transform file metadata into cloud-native schematics. And develop a migration strategy that includes effective file conversion techniques.

Potential use cases

On-premises file replication and synchronization are essential for various use cases:

- Downstream or upstream dependencies, like when applications that run on a mainframe and applications that run on Azure need to exchange data via files
- Parallel testing of rehosted or re-engineered applications on Azure with on-premises applications
- Tightly coupled on-premises applications on systems that can't be immediately remediated or modernized

Contributors

Microsoft maintains this article. The following contributors wrote this article.

Principal authors:

- [Nithish Aruldoss](#) | Engineering Architect
- [Ashish Khandelwal](#) | Principal Engineering Architecture Manager

Other contributors:

- [Gyani Sinha](#) | Senior Cloud Solution Architect

To see nonpublic LinkedIn profiles, sign in to LinkedIn.

Next steps

- For more information, contact the [Microsoft SQL Data Engineering team](#).
- [Azure database migration guides](#)

Related resources

- [Replicate and sync mainframe data in Azure](#)
- [Modernize mainframe and midrange data](#)
- [Migrate IBM mainframe applications to Azure with TmaxSoft OpenFrame](#)
- [Unisys mainframe migration with Avanade Automated Migration Technology](#)

Modernize mainframe workloads by using BMC AMI Cloud

Azure Blob Storage

Azure ExpressRoute

Azure VPN Gateway

Azure Synapse Analytics

Azure Monitor

Organizations can modernize their mainframe systems to take advantage of the benefits of cloud computing. The integration of mainframe data with cloud platforms enhances scalability, performance, and cost efficiency.

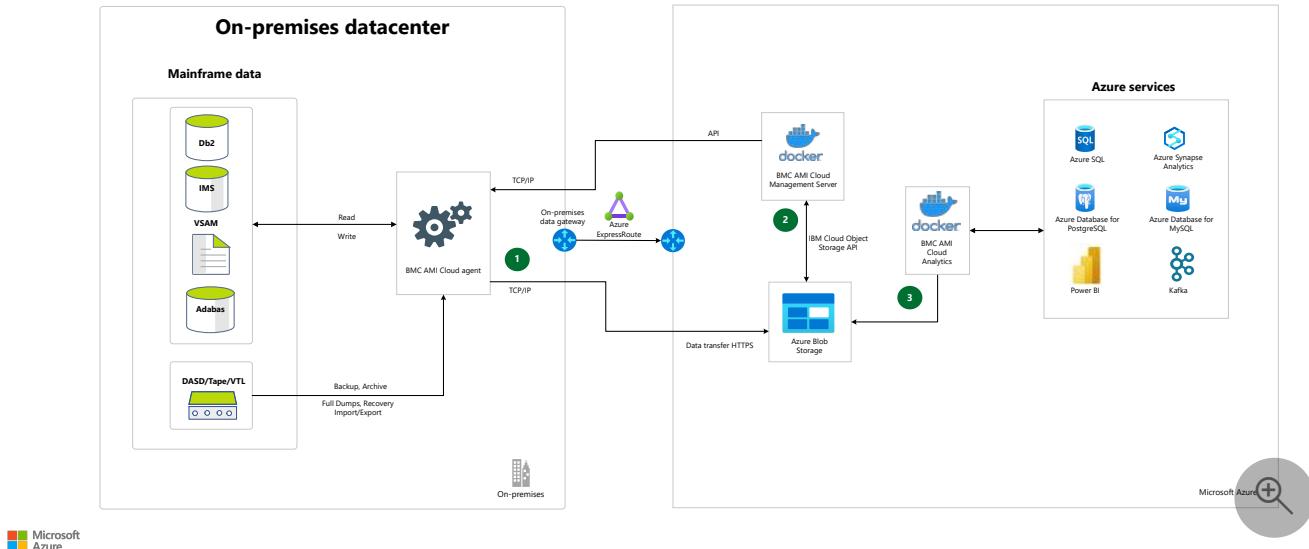
BMC AMI Cloud provides a solution that transfers mainframe data directly to Azure Blob Storage. This service streamlines the migration and modernization journey for organizations.

Apache®, [Kafka](#), and the flame logo are either registered trademarks or trademarks of the Apache Software Foundation in the United States and/or other countries. No endorsement by The Apache Software Foundation is implied by the use of these marks.

Key benefits

- **Cost-effective backup.** Use BMC AMI Cloud and Blob Storage as an efficient alternative to virtual tape libraries to help ensure faster and more economical data backups. This shift reduces costs and improves backup and recovery times, which are essential for business continuity.
- **Data transformation.** BMC AMI Cloud Analytics converts mainframe data into open formats that are compatible with various Azure services. Open formats enhance data usability and integration. This process is crucial for organizations that aim to use advanced analytics, AI, and machine learning tools on their legacy data.
- **Data protection.** BMC AMI Cloud Vault provides immutable, air-gapped copies of mainframe data in Azure storage. It protects data by providing versioning, locking, immutability, and encryption. It also provides threat protection and complies with regulatory requirements for data retention.

Architecture



Download a [Visio file](#) of this architecture.

Workflow

The architecture of BMC AMI Cloud integration with Azure includes several components. Each component plays an important role in the data migration and transformation process.

1. The BMC AMI Cloud agent starts a z/OS task that sends encrypted and compressed mainframe data to Blob Storage over Transmission Control Protocol/Internet Protocol (TCP/IP). This process helps ensure secure and efficient data transfer without the need for intermediate storage, which reduces latency and potential points of failure.
2. BMC AMI Cloud Management Server, a Docker-based web application, administers the cloud agents. It manages policies, activities, and storage, which helps ensure seamless data management.
3. BMC AMI Cloud Analytics converts mainframe data that's stored in Blob Storage into formats that are suitable for AI, business intelligence, and machine learning applications. BMC AMI Cloud Analytics supports conversion to CSV and JSON, and enables direct integration with Azure Databases. This capability supports a wide range of analytical and operational use cases.

Components

Each component of BMC AMI Cloud Data is designed to optimize various aspects of the data migration and management process:

- BMC AMI Cloud agent is a Java-based application that runs as a started task on one or more z/OS logical partitions (LPARs). It reads and writes data directly to and from Blob

Storage over TCP/IP. The BMC AMI Cloud agent uses the zIIP engine, which significantly reduces general CPU consumption. This optimization enhances mainframe performance and lowers cost. You can use multiple agents to increase scalability and resilience. In this architecture, BMC AMI Cloud agent serves as the primary data transfer mechanism that securely moves mainframe data to Azure Storage.

- BMC AMI Cloud Management server is a web application that runs in a Docker container that manages the web UI and communication with z/OS agents. It provides a way to define policies for data protection, data migration, and data archival. These policies help ensure that data management aligns with organizational requirements and compliance standards. For superior availability, deploy this application on Azure Virtual Machines within your virtual network. In this architecture, BMC AMI Cloud Management server acts as the central control plane for managing data migration policies and monitoring agent activities.
- The lifecycle management engine is a Java-based application that runs on-premises on a z/OS LPAR. It deletes expired data from object storage and z/OS. This process automates data lifecycle management and helps ensure that storage resources are used efficiently. In this architecture, the lifecycle management engine automates data retention policies to optimize storage costs and compliance.
- The data management command-line interface (CLI) is a CLI that runs on z/OS LPARs. Users can perform backup, restore, archive, recall, and delete actions to and from Blob Storage by using the CLI. The CLI provides flexibility and control over data management tasks and enables integration with existing workflows and scripts. In this architecture, the data management CLI provides administrators with direct command-line control over data operations between mainframe and Azure Storage.
- BMC AMI Cloud Analytics is a Docker-based application that transforms BMC AMI Cloud-managed objects into open formats that AI, business intelligence, and machine learning applications can process. This capability allows organizations to unlock the value of their mainframe data by making it accessible to modern analytical tools. In this architecture, BMC AMI Cloud Analytics enables integration of mainframe data with Azure analytics services by converting data into formats compatible with modern data platforms.

Networking and identity

Secure and reliable connectivity between on-premises mainframe systems and Azure cloud services is crucial for the success of any modernization effort.

- [Azure ExpressRoute](#) is a connectivity service that provides a private and reliable connection to Azure services. This connection delivers superior performance and

enhanced security compared to public internet connections. In this architecture, ExpressRoute helps you transfer mainframe data from on-premises environments to Azure via a private connection. This approach is ideal for organizations that have stringent data sovereignty requirements.

- [Azure VPN Gateway](#) is a virtual network gateway that sends encrypted traffic between Azure Virtual Network and on-premises locations over the public internet. In this architecture, you can deploy VPN Gateway for scenarios where you can't use a dedicated private connection to transfer the mainframe data to Azure.
- [Microsoft Entra ID](#) is an identity and access management service that can synchronize with on-premises directories. Microsoft Entra ID supports single sign-on and multifactor authentication, which enhances security and user experience. In this architecture, Microsoft Entra ID helps ensure secure authentication and access control for BMC AMI Cloud components, and you can manage and administer the permissions by using Azure role-based access control (Azure RBAC).

Databases and storage

The mainframe data is migrated to Azure Storage through the BMC AMI Cloud agent. You can integrate the data in Storage with any of the following Azure database services by using BMC AMI Cloud Analytics.

- [Azure Database for PostgreSQL](#) is a fully managed, relational database service that's based on the community edition of the open-source PostgreSQL database engine. You can use Azure Database for PostgreSQL to focus on application innovation instead of database management. You can also scale your workload quickly and easily. In this architecture, you can integrate mainframe data with Azure Database for PostgreSQL through BMC AMI Cloud Analytics.
- [Azure Database for MySQL](#) is a fully managed, relational database service that's based on the community edition of the open-source MySQL database engine. In this architecture, you can integrate mainframe data with Azure Database for MySQL through BMC AMI Cloud Analytics.
- [Azure SQL Database](#) is a fully managed, scalable database service that has AI-powered features for performance and durability optimization. It supports serverless compute and Hyperscale storage options and automatically scales resources on demand. In this architecture, you can integrate mainframe data in Storage with SQL Database by using BMC AMI Cloud Analytics.
- [Azure SQL Managed Instance](#) is an intelligent, scalable cloud database service that provides all the benefits of a fully managed and evergreen platform as a service. SQL

Managed Instance provides near-complete compatibility with the latest SQL Server (Enterprise Edition) database engine. This service also provides a native virtual network implementation that addresses common security concerns. In this architecture, you can integrate mainframe data with SQL Managed Instance through BMC AMI Cloud Analytics.

- [Azure Synapse Analytics](#) is a fast and flexible cloud data warehouse that helps you scale, compute, and store data elastically and independently, with a massively parallel processing architecture. In this architecture, you can integrate mainframe data with Azure Synapse Analytics for advanced analytics and data warehousing through BMC AMI Cloud Analytics.
- [Storage](#) is a comprehensive cloud storage solution that includes object, file, disk, queue, and table storage. Storage supports hybrid storage solutions and provides tools for data transfer, sharing, and backup. It also provides scalable backup and archival solutions for the migrated mainframe data. In this architecture, Storage serves as the primary destination for mainframe data transferred by BMC AMI Cloud agents.

Analysis and monitoring

Effective monitoring and analysis are essential for maintaining the health and performance of cloud-based systems:

- [Azure Monitor](#) is a comprehensive monitoring service that provides a solution for collecting, analyzing, and acting on telemetry from cloud and on-premises environments. It includes features such as Application Insights, Azure Monitor Logs, and Log Analytics. These features enable proactive monitoring and problem resolution. In this architecture, you can monitor and analyze the metrics during data migration from the mainframe to Storage by using Azure Monitor.
- [Power BI](#) is a group of business analytics tools that connect to hundreds of data sources, which simplifies data preparation and drives unplanned analysis. In this architecture, Power BI can access the migrated data in Storage or Azure databases to create interactive reports that provide insights and dashboards.

Implementation alternatives

You can choose between on-premises and cloud deployment options based on your specific needs and constraints:

- **On-premises deployment.** You can install BMC AMI Cloud Management Server on-premises on z/OS Container Extensions (zCX) or on a Linux virtual instance. This

installation provides flexibility for organizations that have regulatory or latency requirements.

- **Data transformation service.** BMC AMI Cloud Analytics can operate externally to the mainframe in an on-premises environment. It can also be deployed in the cloud by using server instances or container services, which enhances resource usage and performance.

Use cases

This architecture is suitable for various use cases.

- **Mainframe data accessibility:** Make mainframe data available to Azure data services, AI, machine learning, analytics, and business intelligence tools.
- **Data protection:** Back up and archive mainframe data to Blob Storage to help ensure data availability and durability.
- **Direct data integration:** Enable mainframe applications to write and read data directly to and from Blob Storage to streamline workflows and reduce latency.
- **Cybersecurity:** Protect mainframe data against cyberattacks by creating an immutable third copy in Azure to enhance data security and compliance.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Seetharaman Sankaran](#) | Senior Engineering Architect

Other contributors:

- [Pratim Dasgupta](#) | Senior Engineering Architect
- [Ashish Khandelwal](#) | Principal Engineering Architect Manager

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

- [BMC AMI Cloud data management for mainframe](#)
- [What is ExpressRoute?](#)
- [What is VPN Gateway?](#)

- [Introduction to Azure Data Lake Storage Gen2](#)
- [What is SQL Database?](#)
- [What is Azure Synapse Analytics?](#)
- [What is Azure Database for PostgreSQL?](#)
- [What is Azure Database for MySQL?](#)
- [What is Power BI?](#)
- [Azure Monitor overview](#)
- For more information, contact the [Mainframe Data Modernization team](#)

Related resources

- [Modernize mainframe and midrange data](#)
- [Re-engineer mainframe batch applications on Azure](#)
- [Replicate and sync mainframe data in Azure](#)
- [Mainframe file replication and sync on Azure](#)

Move archive data from mainframe systems to Azure

Azure Data Factory

Azure Storage

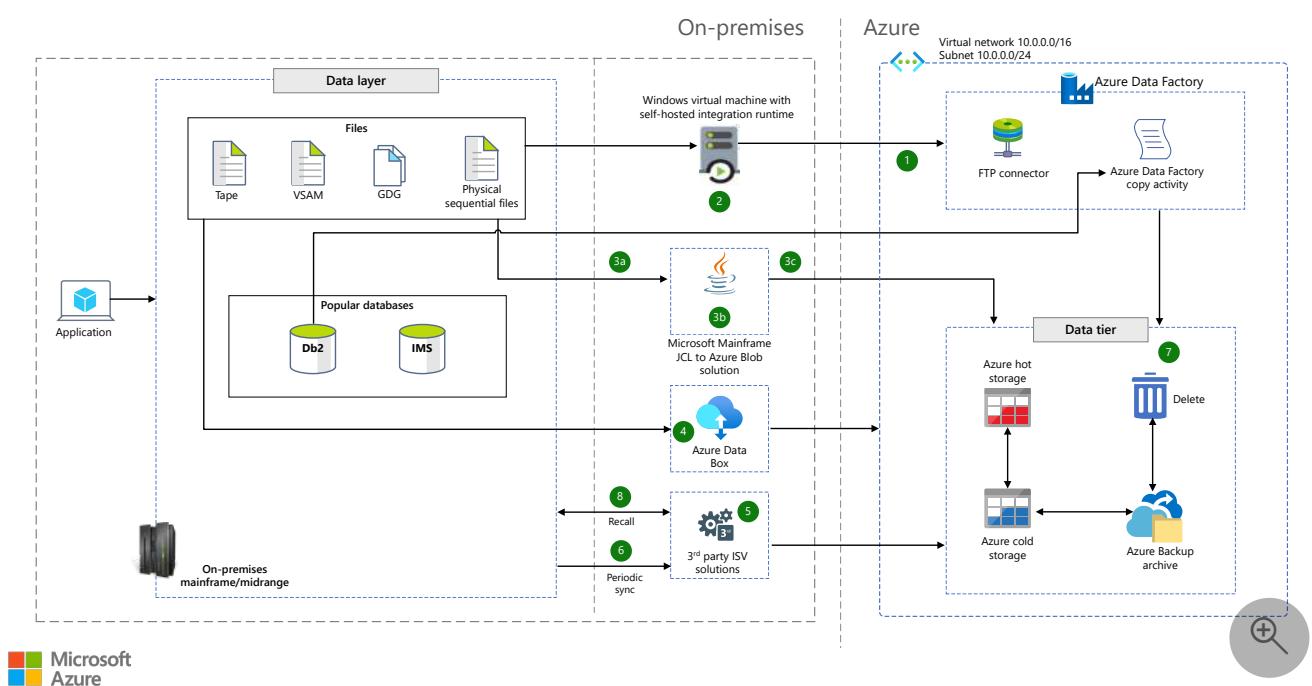
Azure Files

Azure Blob Storage

Azure Data Box

This reference architecture shows how to move data from mainframe and midrange systems to Azure. In this architecture, archived data is serviced and used only in the mainframe system. Azure is used only as a storage medium.

Architecture



Download a [Visio file](#) of this architecture.

To decide which method to use for moving data between the mainframe system and Azure storage, consider the frequency of data retrieval and the amount of data. Microsoft and third-party solutions are available:

- **Microsoft solutions.**
 - The Azure Data Factory FTP connector.
 - The Data Factory copy activity, which can copy data to any Azure storage solution.
 - *Mainframe JCL to Azure Blob using Java*, a custom solution for moving data from the mainframe system to Azure via Job Control Language (JCL). For more information, contact datasqlninja@microsoft.com.
- **Third-party archive solutions.** Solutions that you can easily integrate with mainframe systems, midrange systems, and Azure services.

Workflow

1. The Azure Data Factory [FTP connector](#) moves data from the mainframe system to Azure [Blob Storage](#). This solution requires an intermediate virtual machine (VM) on which a self-hosted integration runtime is installed.
2. The Data Factory [copy activity](#) connects to the [Db2 database](#) to copy data into Azure [storage](#). This solution also requires an intermediate VM on which a self-hosted integration runtime is installed.
3. The Microsoft *Mainframe JCL to Azure Blob using Java* custom solution moves data between the mainframe system and Blob Storage, and vice versa. This solution is based on Java and runs on Unix System Services on the mainframe. You can get this solution by contacting datasqlninja@microsoft.com.
 - a. You need to complete a one-time configuration of the solution. This configuration involves getting the Blob Storage access keys and moving required artifacts to the mainframe system.
 - b. A JCL submission moves files to and from the mainframe and Blob Storage.
 - c. Files are stored in binary format on Azure. You can configure the custom solution to convert EBCDIC to ASCII for simple data types.
4. Optionally, Azure Data Box can help you physically transfer mainframe data to Azure. This option is appropriate when a large amount of data needs to be migrated and online methods of transmission take too long. (For example, if migration takes weeks.)
5. Easy interaction with the mainframe or midrange environment is provided by [third-party archive solutions](#).

These solutions interact with the mainframe and handle various mainframe parameters, like data types, record types, storage types, and access methods. They serve as a bridge between Azure and the mainframe. Some third-party solutions connect a storage drive to the mainframe and help transfer data to Azure.
6. Data is periodically synced and archived via the third-party archive solution. After the data is available via the third-party solution, the solution can easily push it to Azure by using available connectors.
7. Data is [stored in Azure](#).
8. As needed, [data is recalled from Azure](#) back to the mainframe or midrange systems.

Components

- [Azure Data Factory](#) is a cloud-based hybrid data integration service that you can use to create, schedule, and orchestrate your extract, transform, load (ETL) and extract, load, transfer (ELT) workflows. In this architecture, Azure Data Factory orchestrates the movement of data from mainframe systems to Azure storage by using FTP connectors and copy activities.
- [Azure Files](#) is a cloud storage service that provides simple and secure serverless cloud file shares. These components are used for synchronization and data retention. In this architecture, Azure Files enables file-based data archiving and provides NFS/SMB access for mainframe systems to store and retrieve archived data.
- [Azure storage](#) is a cloud platform that provides scalable, secure cloud storage for your data, apps, and workloads. In this architecture, Azure storage serves as the primary destination for archived mainframe data and provides cost-effective, long-term storage and lifecycle management capabilities.
- [Data Box](#) is a physical device that you can use to move on-premises data to Azure. In this architecture, Data Box provides an option for physically transferring large volumes of mainframe data to Azure when online methods take too long.

Alternatives

You can use the classic method of moving the data out of the mainframe or midrange system via FTP. Data Factory provides an [FTP connector](#) that you can use to archive the data on Azure.

Scenario details

Mainframe and midrange systems generate, process, and store huge amounts of data. When this data gets old, it's not typically useful. However, compliance and regulatory rules sometimes require this data to be stored for a certain number of years, so archiving it is critical. By archiving this data, you can reduce costs and optimize resources. Archiving data also helps with data analytics and provides a history of your data.

Potential use cases

Archiving data to the cloud can help you:

- Free up storage resources in mainframe and midrange systems.
- Optimize performance for queries by storing only relevant data on the active system.
- Reduce operational costs by storing data in a more economical way.

- Use archived data for analytics to create new opportunities and make better business decisions.

Recommendations

Depending on how you use data, you might want to convert it to ASCII from binary and then upload it to Azure. Doing so makes analytics easier on Azure.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that can be used to improve the quality of a workload. For more information, see [Microsoft Azure Well-Architected Framework](#).

- Complex data types on the mainframe must be handled during archive.
- Application subject matter experts can identify which data needs to be archived.
- To determine the amount of time between syncs, consider factors like business criticality, compliance needs, and frequency of data access.

Cost Optimization

Cost Optimization is about looking at ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

Use the Azure [pricing calculator](#) to estimate the cost of implementing this solution.

Third-party archive solutions

Some third-party solutions are available on [Azure Marketplace](#). Each of these solutions requires unique configuration. Setting up these solutions is one of the primary tasks of implementing this architecture.

Azure storage

Azure has a variety of options for different application and technical requirements, like frequent versus infrequent access, and structured versus unstructured data. You can set up various storage lifecycle configurations in Azure storage. You can define the rules to manage the lifecycle. For an overview, see [Configure a lifecycle management policy](#).

Data recall

Recall of archived data is an important aspect of archive solutions. Few of the third-party solutions provide a seamless experience for recalling archived data. It's as simple as running a command on-premises. The third-party agent automatically gets the data from Azure and ingests it back into the mainframe system.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Pratim Dasgupta](#) | Engineering Architect

Other contributors:

- [Ashish Khandelwal](#) | Senior Engineering Architect Manager
- [Ramanath Nayak](#) | Engineering Architect

Next steps

For more information, contact [Azure Data Engineering - Mainframe/Midrange Modernization](#).

See these resources:

- [Azure Database Migration Guides](#)
- [What is Azure Data Factory?](#)
- [Introduction to Azure Storage](#)
- [What is Azure Files?](#)
- [What is Azure Data Box?](#)
- [Explore Azure Storage services](#)

Related resources

- [Modernize mainframe and midrange data](#)
- [Re-engineer IBM z/OS batch applications on Azure](#)
- [Replicate and sync mainframe data in Azure](#)

Virtual desktop architecture design

07/30/2025

Migrating end-user desktops to the cloud helps improve employee productivity and enables employees to work from anywhere on a high-security cloud-based virtual desktop infrastructure.

Azure provides these virtual desktop solutions:

- [Azure Virtual Desktop](#) is a desktop and application virtualization service.
- [Omniassa Horizon Cloud on Microsoft Azure](#) is an Omniassa service that simplifies the delivery of virtual desktops and applications on Azure by extending Azure Virtual Desktop.
- [Citrix Virtual Apps and Desktops for Azure](#) is a desktop and app virtualization service that you can use to provision Windows desktops and apps on Azure with Citrix and Azure Virtual Desktop.
- [Microsoft Dev Box](#) is a service that gives developers access to ready-to-code, project-specific workstations that are preconfigured and centrally managed in the cloud.

Introduction to virtual desktop architecture on Azure

If you're new to virtual desktops on Azure, the best way to learn more is [Microsoft Learn training](#), a free online platform. Here's a learning path to get you started:

- [Deliver remote desktops and apps with Azure Virtual Desktop](#)

Path to production

Cloud Adoption Framework for Azure provides an end-to-end scenario to guide you through your virtual desktop migration or deployment. Start with [Migrate or deploy Azure Virtual Desktop instances to Azure](#), and check out the other articles below that one in the table of contents.

These are two more key Cloud Adoption Framework articles:

- [Azure Virtual Desktop planning](#)
- [Azure Virtual Desktop Azure landing zone review](#)

See [Understanding Azure Virtual Desktop network connectivity](#) for a high-level overview of the network connections used by Azure Virtual Desktop.

Best practices

- [Security best practices for Azure Virtual Desktop](#)
- [Azure security baseline for Azure Virtual Desktop](#)
- [Session host virtual machine sizing guidelines](#)
- [Configure device redirection](#)
- [Set up scaling tool using Azure Automation and Azure Logic Apps for Azure Virtual Desktop](#)

More virtual desktop resources

The following sections, organized by category, provide links to example scenarios and other articles.

Identity

- [Authentication in Azure Virtual Desktop](#)
- [Deploy Microsoft Entra joined virtual machines in Azure Virtual Desktop](#)
- [Compare self-managed Active Directory Domain Services, Microsoft Entra ID, and managed Microsoft Entra Domain Services](#)

Azure Virtual Desktop for the enterprise

- [Azure Virtual Desktop for the enterprise](#)

FSLogix

FSLogix is designed for roaming profiles in remote computing environments like Azure Virtual Desktop. It stores a complete user profile in a single container. At sign-in, this container is dynamically attached to the computing environment. For more information, see these resources:

- [FSLogix configuration examples](#)
- [FSLogix profile containers and Azure Files](#)
- [Storage options for FSLogix profile containers in Azure Virtual Desktop](#)

Stay current with virtual desktop technologies on Azure

Get the latest updates on Azure Virtual Desktop technologies .

Additional resources

Example solutions

These are some additional articles about Azure Virtual Desktop:

- [Azure Virtual Desktop RDP Shortpath for managed networks](#)
- [Multiregion Business Continuity and Disaster Recovery \(BCDR\) for Azure Virtual Desktop](#)
- [Deploy Esri ArcGIS Pro in Azure Virtual Desktop](#)

AWS professionals

- [AWS to Azure services comparison - End-user computing](#)

Azure Virtual Desktop landing zone design guide

06/28/2025

This article provides an overview of the design areas in the [Azure landing zone architecture for Azure Virtual Desktop](#). It targets architects and technical decision-makers. Use this guidance to quickly gain an understanding of the Virtual Desktop landing zone reference implementation.

Landing zone concepts

An [Azure landing zone](#) is an environment that follows key design principles across eight design areas. These design principles accommodate all application portfolios and enable application migration, modernization, and innovation at scale. An Azure landing zone uses subscriptions to isolate and scale application resources and platform resources.

An Azure landing zone provides the necessary foundation for cloud workloads such as Virtual Desktop. It defines essential components like governance, security, networking, identity, and operations. To host and manage services at scale, you need all these components.

Types of landing zones

Azure landing zones have [two categories](#):

- **Platform landing zones** provide shared foundational services like networking, identity management, and resource governance. Platform landing zones form the core infrastructure that supports application workloads.
- **Application landing zones** host specific applications, workloads, or services. They provide the necessary environment to run applications. Policies and management groups enforce governance in application landing zones.

Reference architecture

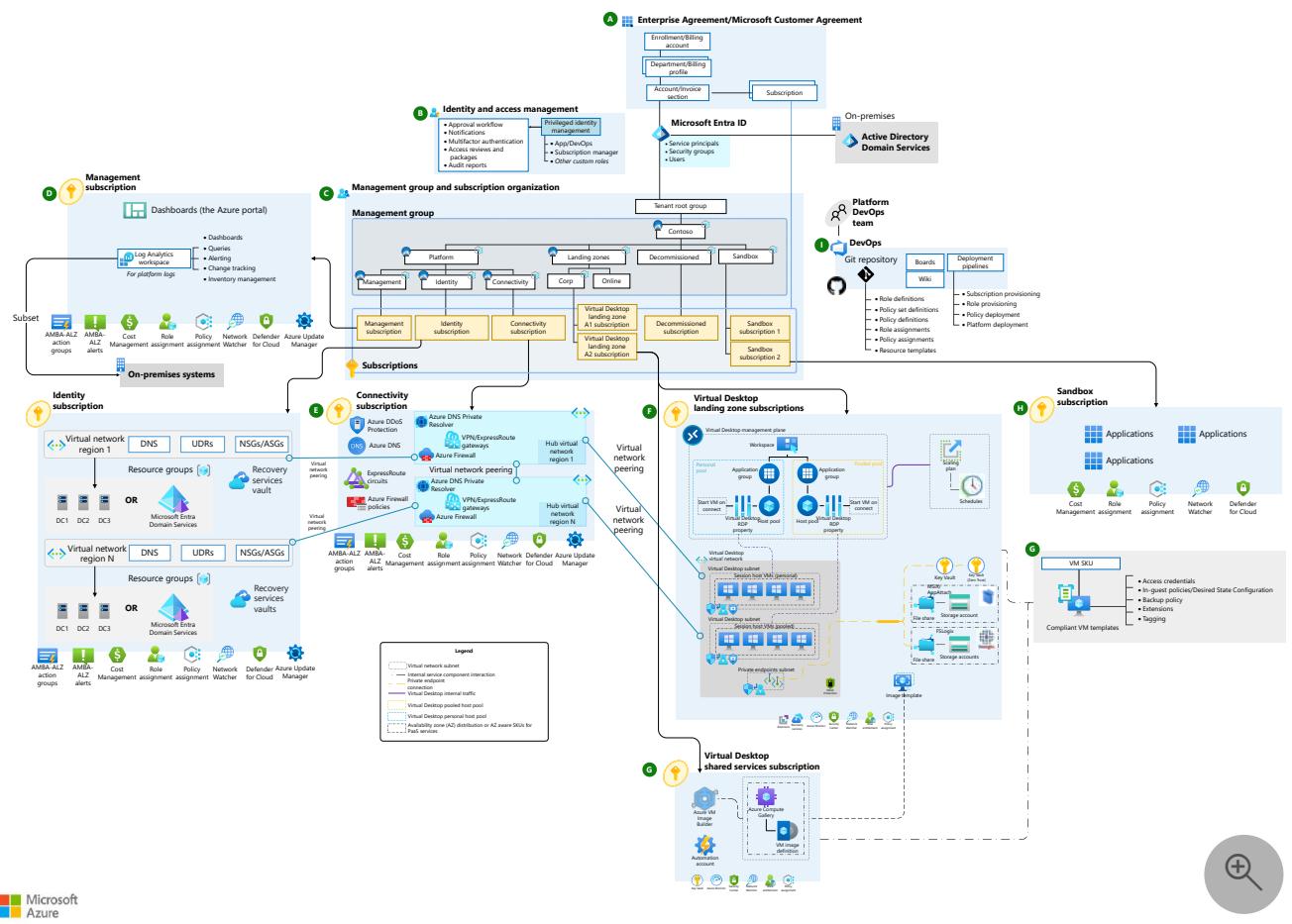
The Virtual Desktop reference architecture is a proven architecture for running Virtual Desktop in an application landing zone. Start with this architecture for Virtual Desktop deployments.

The [landing zone architecture for Virtual Desktop](#) is part of the [Virtual Desktop scenario article series](#) in the Cloud Adoption Framework for Azure. This series describes compatibility requirements, design principles, and deployment guidance for the landing zone.

When you design Virtual Desktop to run from an application landing zone, follow this structured architecture to ensure scalability, security, and operational excellence. This architecture provides a robust foundation to deploy Virtual Desktop at scale while maintaining centralized governance, security, and performance.

Benefits of this reference architecture

- **Scalability:** Supports large-scale deployments so that you can quickly scale resources based on demand
- **Security:** Uses security measures like Azure role-based access control (Azure RBAC) and network security to protect your environment from threats
- **Operational efficiency:** Includes automation and monitoring tools to reduce operational burden and improve system performance



Download a [Visio file](#) of this architecture.

Design areas

The letters "A" through "I" in the diagram indicate design areas for the Virtual Desktop landing zone. This diagram shows the hierarchy of resource organization.

Legend	Design area	Objective
A	Azure billing and Active Directory tenant	Set up the tenant, enrollment, and billing configuration early on.
B	Identity and access management	Establish secure access to Virtual Desktop through Microsoft Entra ID, conditional access, and Azure RBAC. Identity and access management is a primary security boundary in the public cloud. It serves as the foundation of secure and fully compliant architectures.
C	Resource organization	Design subscriptions and management group hierarchies to support governance, operations management, and adoption patterns at scale.
D, G, H	Management and monitoring	Create a baseline that ensures operational visibility, compliance, and the ability to protect and recover workloads.
E	Network topology and connectivity	Design reliable and scalable network architecture as a foundational element of the cloud environment.
F	Security	Apply security controls directly within the Virtual Desktop workload.
G, F	Business continuity and disaster recovery	Create business continuity and disaster recovery strategies for Virtual Desktop and its supporting services to ensure resilience and recovery. This area isn't explicitly labeled in the diagram but is represented within sections G and F.
I	Platform automation and DevOps	Enable infrastructure as code and continuous integration and continuous delivery (CI/CD) pipelines to manage platform-level resources, management group policies, role definitions, and subscription provisioning.

💡 Tip

Review the Virtual Desktop design areas to ensure alignment with Virtual Desktop best practices.

Virtual Desktop Azure landing zone design areas: These areas define the foundational elements required to set up the Virtual Desktop deployment in an enterprise-scale environment. They focus on preparing resources like network configurations, identity management, security, and governance. This landing zone helps create a scalable and secure environment for Virtual Desktop workloads.

Virtual Desktop design areas: These areas represent the architectural principles and best practices for designing and operating Virtual Desktop workloads. They cover aspects such as application delivery, infrastructure design, security, and cost optimization. Each area

aligns with Azure best practices to ensure an optimal and cost-effective Virtual Desktop implementation.

Design principles

Like other landing zones, the Virtual Desktop landing zone follows the core [Azure landing zone design principles](#) and aligns with common [design areas](#).

This architecture uses the following key principles:

- **Subscription democratization:** Teams manage their own resources within a controlled framework.
- **Policy-driven governance:** Centralized policies and controls enforce compliance and governance.
- **Application-focused service model:** The architecture supports organizations structured around applications.
- **Single control and management plane:** Centralized resource management maintains oversight and control.

Reference implementation

The Virtual Desktop application landing zone reference implementation follows the reference architecture and design principles outlined in the Cloud Adoption Framework and the Azure Well-Architected Framework. This solution provides steps to prepare landing zone subscriptions for a scalable Virtual Desktop deployment and to deploy Virtual Desktop within those landing zone subscriptions.

The Virtual Desktop landing zone implementation provides your organization with an enterprise-ready Virtual Desktop deployment that aligns with best practices in scalability, security, and governance.

Architecture

Important

The accelerator deploys resources into the Virtual Desktop application landing zone and shared services landing zone subscriptions.

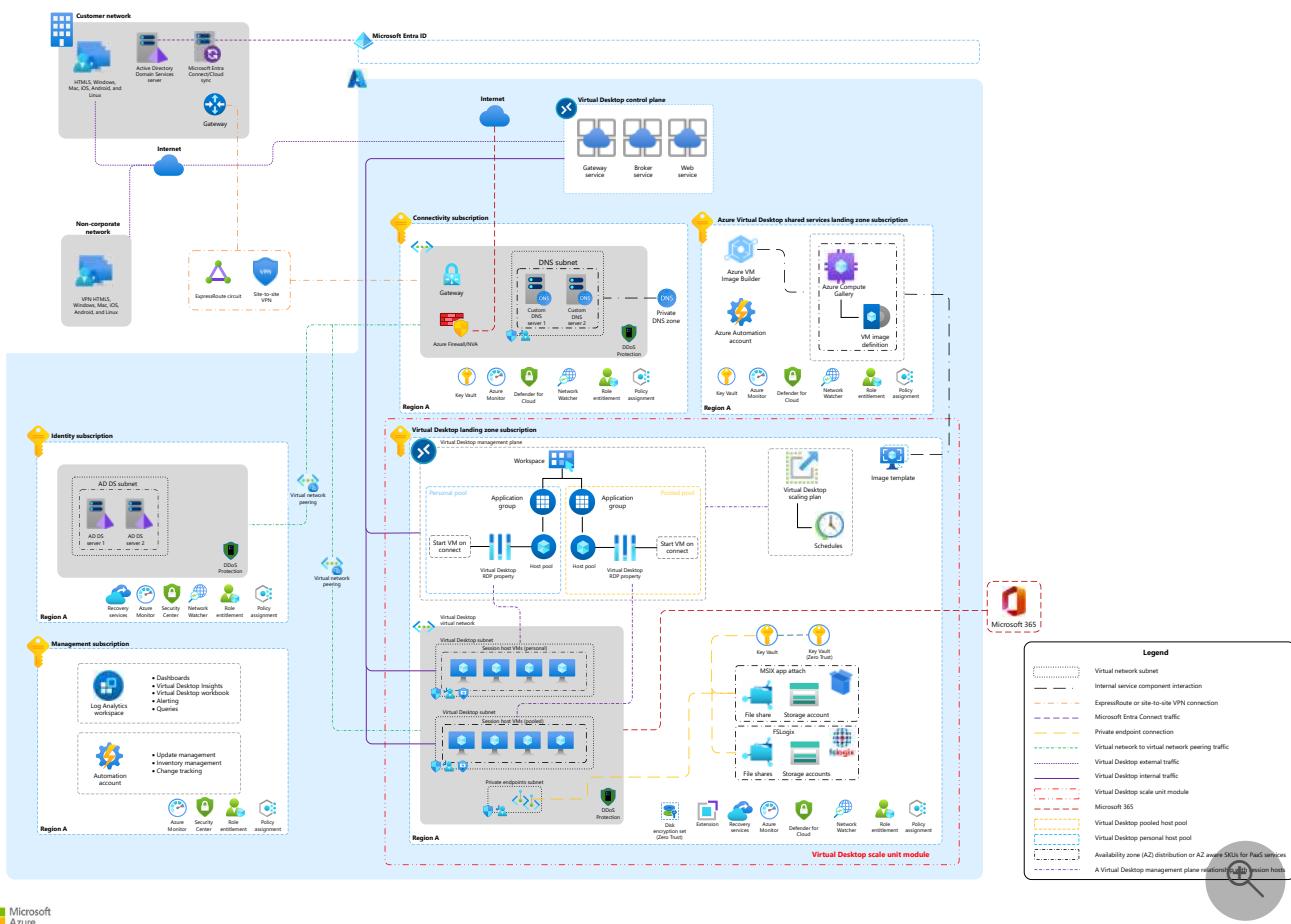
You must deploy a [Cloud Adoption Framework platform landing zone](#) first. This deployment provides the shared foundation services that resources in this implementation require.

Review the Virtual Desktop [prerequisites](#) before you start the [deployment](#).

This architecture is based on multiple subscriptions that are each dedicated to specific purposes:

- **Virtual Desktop subscription:** This subscription, or multiple subscriptions depending on environment scale, deploys the Virtual Desktop resources that are specific to individual workloads, not shared across workloads. These resources include virtual machines, storage accounts, key vaults, and private endpoints. This subscription is considered part of the application landing zone.
- **Virtual Desktop shared services subscription:** This subscription hosts all services shared across multiple Virtual Desktop workloads. It includes resources like Azure Automation accounts, data collection rules, Log Analytics workspaces, and Azure compute galleries. This subscription is considered part of the application landing zone.
- **Platform subscriptions:** These foundational subscriptions provide shared services across the entire environment. They support and connect to application landing zone subscriptions.
 - **Management subscription:** This subscription is part of the Azure landing zone platform structure and typically hosts shared management resources. These resources include monitoring solutions, update management tools, and governance tools. In this architecture, the management subscription isn't an active dependency for the Virtual Desktop workload. The workload team must implement their own automation, monitoring, and management capabilities within their designated workload subscription.
 - **Connectivity subscription:** This subscription contains network-related components like virtual networks, network security groups (NSGs), Azure Firewall, and Azure ExpressRoute or VPN gateways. In this architecture, this subscription provides the Virtual Desktop application landing zone with secure and scalable network infrastructure. This capability enables isolated traffic flows, segmentation between organization workloads, and secure access to cross-premises resources.
 - **Identity subscription:** This subscription handles identity and access management services required to support domain-joined Virtual Desktop session hosts. In this architecture, this subscription provides the Virtual Desktop application landing zone with domain services. These services include Microsoft Entra Domain Services or self-

managed Active Directory domain controllers hosted in Azure. These services enable session hosts to join a domain and authenticate users securely, enforce group policies, and support legacy authentication scenarios that some applications require.



Download a [Visio file](#) of this architecture.

Benefits of this reference implementation

- Scalability:** It scales efficiently to meet your organization's needs.
- Security:** It uses enterprise-grade security, compliance, and governance controls to protect your environment.
- Faster deployment:** It accelerates deployment by using predefined templates, configurations, and best practices.
- Best practices compliance:** It follows the best practices in the architecture.

Accelerator overview



The Virtual Desktop landing zone reference implementation [supports multiple deployment scenarios](#) depending on your requirements. Each deployment scenario supports

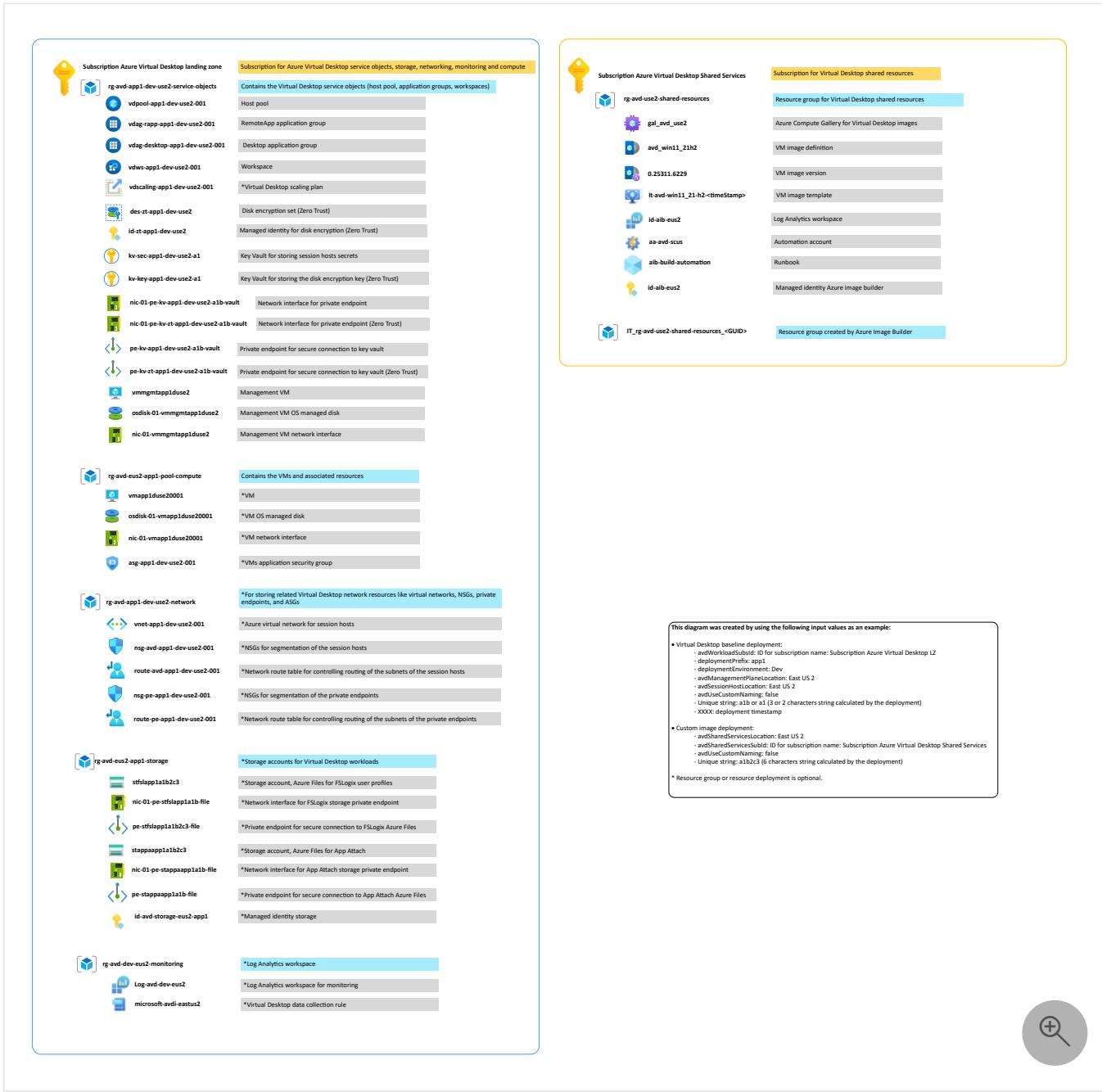
both greenfield and brownfield deployments and provides multiple infrastructure as code (IaC) template options:

- The Azure portal UI
- The Azure CLI or Azure PowerShell Bicep template
- A Terraform template

The accelerator uses resource naming automation based on the following recommendations:

- [Cloud Adoption Framework best practices for naming conventions](#)
- The [recommended abbreviations for Azure resource types](#)
- The [minimum suggested tags](#)

Before you proceed with the deployment scenarios, familiarize yourself with the accelerator's [Azure resource naming, tagging, and organization](#).



Download a [Visio file](#) of the image.

Accelerator deployment

To perform the deployment, do the following steps:

- 1. Review the deployment prerequisites.** This step helps you ready your environment for the deployment.
- 2. Deploy the platform landing zone if you don't already have one.** This step sets up the foundational components.
- 3. Deploy the Virtual Desktop reference implementation.** After the platform landing zone is in place, deploy the Virtual Desktop landing zone reference implementation of the reference architecture.

To start, choose the following deployment scenario that best matches your requirements.

Baseline deployment

The baseline deployment deploys the Virtual Desktop resources and dependent services required to establish a Virtual Desktop baseline.

This deployment scenario includes the following items:

- **Virtual Desktop** resources, including a workspace, scaling plan, host pool, application groups, session host virtual machines, and optionally, private endpoints
- An [Azure Files share](#) integrated with your identity service
- [Azure Key Vault](#) for secret, key, and certificate management
- Optionally, a new [Azure Virtual Network](#) that includes baseline NSGs, application security groups (ASGs), and route tables
- Optionally, Azure Storage account and Key Vault private endpoints and private Domain Name System (DNS) zones

When you're ready for deployment, do the following steps:

1. Follow the [Baseline Deployment Guide](#) for details about prerequisites, planning information, and deployment components.
2. Optionally, review the **Custom image build deployment** tab to build an updated image for your Virtual Desktop host sessions.
3. Do the [baseline deployment steps](#). If you created a custom Azure Compute Gallery image in the previous step, on the **Session hosts** page, choose *Compute Gallery* as the **OS selection source**. Then select the correct **Image**.

Azure Virtual Desktop - Landing Zone Accelerator (LZA) - Baseline

Deploy from a custom template

Deployment Basics Identity Management plane **Session hosts** Storage Networking Monitoring

Deploy session hosts

Region Settings

Session hosts region *

Domain Join Settings

Custom OU path (Optional) Example: OU=session-hosts,OU=avd,DC=contoso,DC=com

OS selection

Source * Compute Gallery

Compute Gallery *

Image Definition * 

Next steps

To build on the concepts from this design guide, explore the following Microsoft Learn resources:

- [Enterprise-scale support for Virtual Desktop landing zone accelerator](#): Learn how to deploy a Virtual Desktop landing zone by using IaC accelerators that align with the Cloud Adoption Framework.
- [Network topology and connectivity for Virtual Desktop](#): Explore recommended network designs, including hub-and-spoke topology, hybrid connectivity, RDP Shortpath, and security best practices for Virtual Desktop.
- [Security, governance, and compliance for Virtual Desktop](#): Understand how to implement security controls, Azure RBAC, monitoring, and governance to ensure the security and compliance of your Virtual Desktop environment.

Windows 365 Azure network connection

Azure ExpressRoute

Azure Virtual Desktop

Azure Virtual Machines

Azure Virtual Network

Windows 365 is a cloud-based service that you can use to deliver highly optimized and personalized Windows computing instances called Cloud PCs that are purpose-built for each user's requirements. Cloud PCs use a combination of the following services:

- Intune to customize, secure, and manage Cloud PCs
- Entra ID for identity and access control
- Azure Virtual Desktop for remote connectivity

A Cloud PC is a highly available, optimized, and personalized computing instance that provides you with a rich Windows desktop experience. It's hosted in the Windows 365 service and is accessible from anywhere, on any device.

The Windows 365 shared responsibility model

Windows 365 is a software as a service (SaaS) solution. Microsoft manages some components in Windows 365 services, and you manage other components. The amount of responsibility you have depends on the architecture pattern you choose for deployment. The responsibilities to manage Windows 365 are split into three parts:

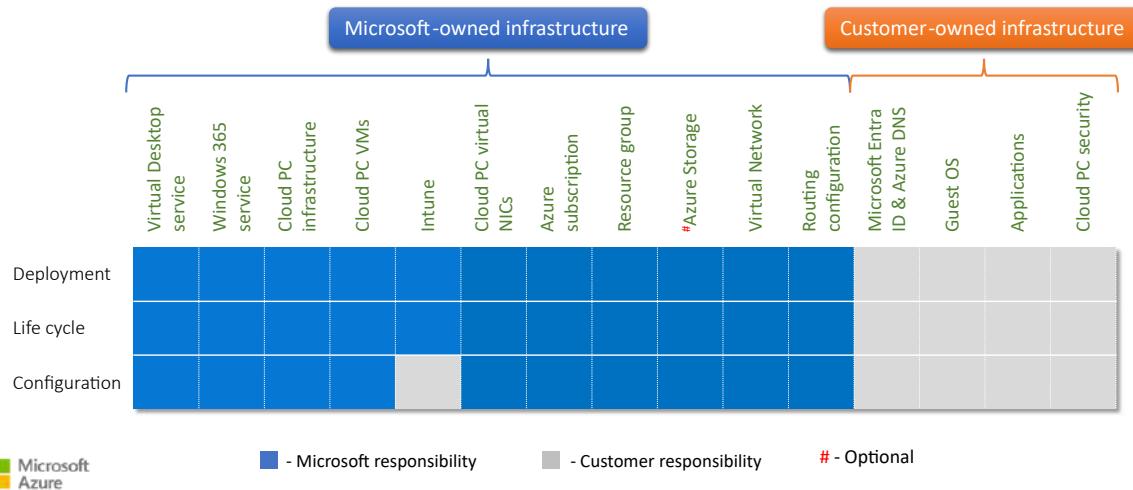
- **Deployment:** Planning and deploying the components of the service.
- **Lifecycle:** Management of the component throughout its lifecycle, such as patching and securing.
- **Configuration:** Configuring the component to apply settings as needed for a scenario.

The following diagram shows the responsibility matrix of a Windows 365 deployment by using the recommended Microsoft-hosted network, Microsoft Entra join, and gallery images with Windows Autopatch. With this configuration, you don't have to manage many components and lifecycle stages. This configuration translates to the benefits listed in [Recommended architecture patterns](#).

Note

The following diagram represents the responsibilities from the infrastructure perspective, such as setting up the hardware and network, and maintaining them. It doesn't include the Windows 365 or Intune tenant subscription setup.

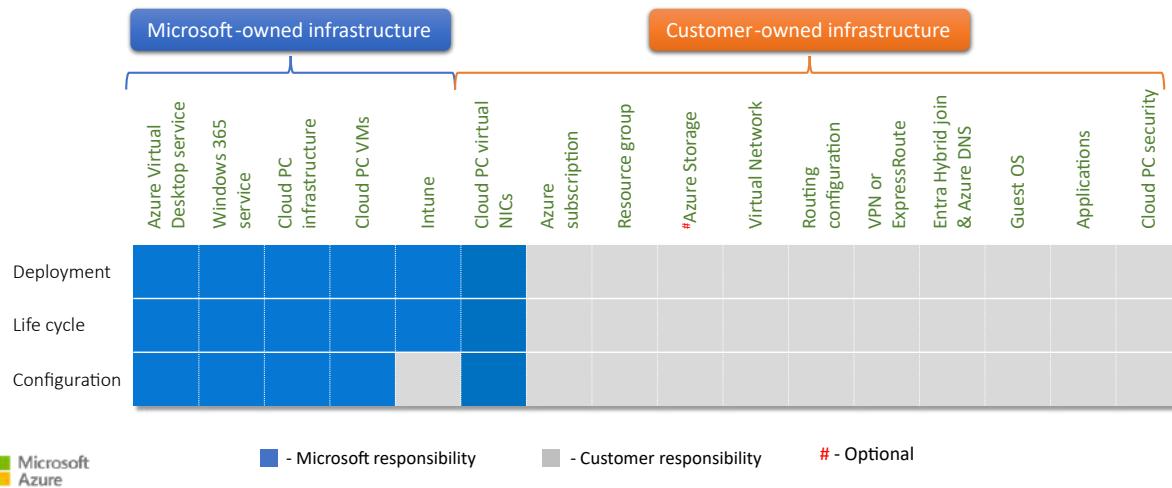
Windows 365 Microsoft-hosted network architecture pattern: Responsibility matrix



[Download a PowerPoint file](#) of this architecture.

The following diagram shows a typical Windows 365 deployment that uses an Azure network connection and shows the components that Microsoft manages and the components that you manage across the lifecycle stages of a Cloud PC.

Windows 365 Azure network connection architecture pattern: Responsibility matrix



[Download a PowerPoint file](#) of this architecture.

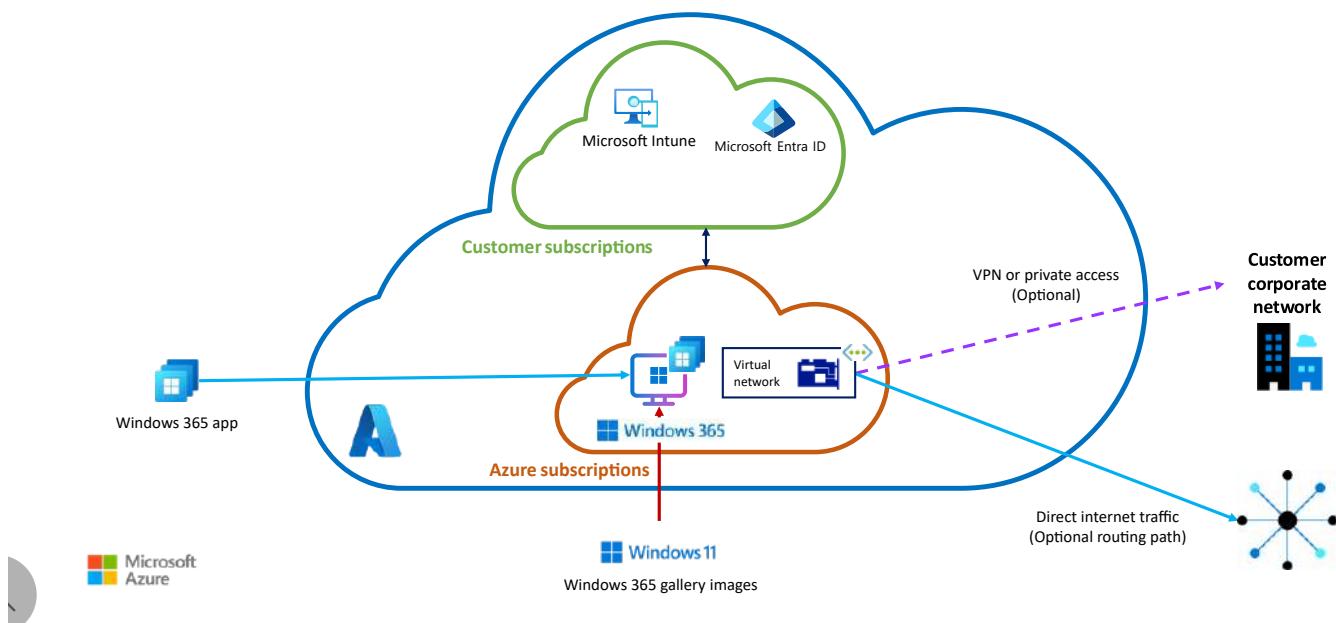
Recommended architecture pattern

Microsoft recommends deploying Windows 365 with the following components to get a SaaS experience, enabling you to have the maximum benefits of the service:

- Microsoft Entra join
- A Microsoft-hosted network
- Gallery images
- Intune-based mobile device management (MDM) service with app and OS configuration
- A Windows 365 app for Cloud PC access

Microsoft-hosted network - Cloud PC's Azure virtual network that's fully managed by Microsoft

Microsoft Entra join



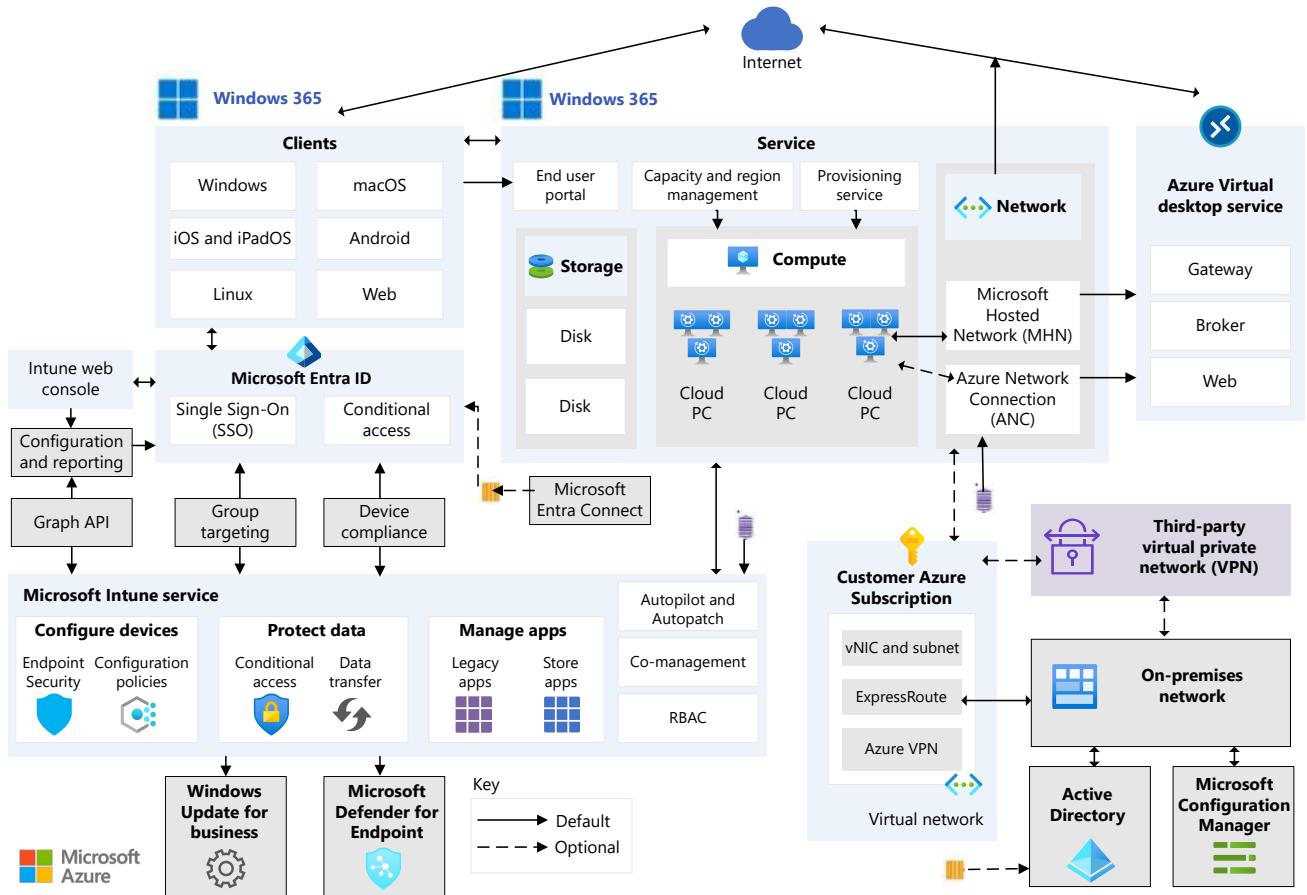
Download a [PowerPoint file](#) of this architecture.

The preceding architecture pattern allows you to get the most out of the Windows 365 service and provides the following benefits:

- Simplified and faster deployment
- Minimal to zero dependencies
- Full Zero Trust framework support
- Simplified troubleshooting flows
- Self-service user troubleshooting
- Low overhead and management
- Highest maturity model of software and application delivery

The Windows 365 service architecture

The following diagram is a representation of all the components that are part of the Windows 365 service. This architecture uses Intune and Microsoft Entra ID, which are core requirements of Windows 365. There are also optional components such as Azure Virtual Network.



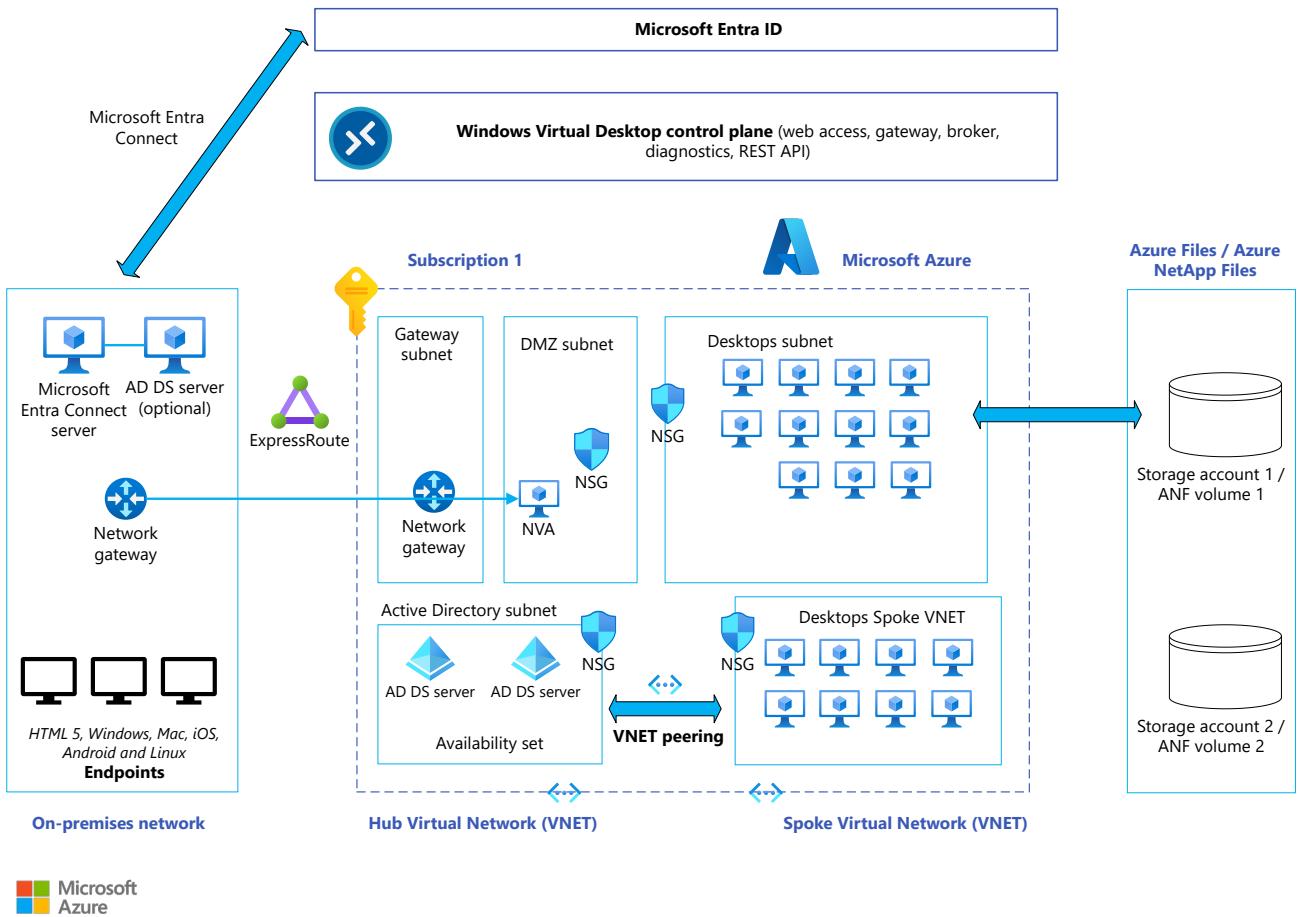
Download a [Visio file](#) of this architecture.

The previous diagram shows Azure network connection and Microsoft-hosted network options. They're mutually exclusive architecture options. The following sections elaborate on the Azure network connection options.

Virtual Desktop

Virtual Desktop is an Azure-based virtual desktop infrastructure (VDI) solution. Microsoft manages Virtual Desktop. It provides a platform-as-a-service (PaaS) style solution. Windows 365 uses the network management components required for you to connect to their Cloud PCs. Components include the Virtual Desktop gateway service, a connection broker service, and a web client service. These services allow for seamless connection to Windows 365 Cloud PCs.

For more information, see [Azure Virtual Desktop for the enterprise](#).



Download a [Visio file](#) of this architecture.

! Note

Windows 365 utilizes the components entitled "Windows Virtual Desktop Control Plane" in the previous diagram for facilitating user and Cloud PC connections and as such inherits most of the connection related capabilities of Azure Virtual Desktop. Familiarizing with how Virtual Desktop networking operates then becomes essential to designing the Azure network connection architecture detailed in this document.

Microsoft Intune

Intune is a cloud-based endpoint management solution that allows you to view and consume reports and manage:

- App delivery
- Windows updates
- Device management configurations
- Security policies

Intune simplifies app and device management across many devices, including mobile devices, desktop computers, and virtual endpoints.

You can protect access and data on organization-owned and personal devices. Intune also has compliance and reporting features that support the Zero Trust security model. For more information, see [Create a device configuration profile](#).

Architecture patterns

An architecture pattern describes components and illustrates the configurations with which a service or product is deployed. For more information, see [Hosted on behalf of architecture](#).

See the following Azure network connection patterns:

Azure network connection with Microsoft Entra join – In this pattern, Microsoft Entra joined Cloud PCs use Azure network connection to connect to resources in on-premises environments, such as line-of-business (LOB) applications, file shares, and other applications that don't need Kerberos or Windows New Technology LAN Manager (NTLM) authentication.

Azure network connection with Microsoft Entra hybrid join – In this pattern, Microsoft Entra hybrid joined Cloud PCs use Azure network connection to domain join with an on-premises Microsoft Entra ID domain controller. The Cloud PC authenticates with the on-premises domain controller when users access the Cloud PC, on-premises apps, or cloud apps that need Kerberos or NTLM authentication.

Azure network connection architecture patterns

For some patterns, the Windows 365 service connects to on-premises environments via Virtual Network by using Azure ExpressRoute or a site-to-site VPN. This connectivity method is represented by Azure network connection, which is an Intune object. This connection allows the Cloud PCs to connect to on-premises resources such as Active Directory or LOB apps.

This network connection, that's represented by Azure network connection, is used by the Windows 365 service during Cloud PC provisioning for on-premises Microsoft Entra domain joining, [health checks](#) for Cloud PC provisioning readiness.

The following tables list dependencies for Azure network connection. Windows 365 runs automatic health checks on these dependencies.

 [Expand table](#)

Dependency	Microsoft Entra Connect - Checks if Microsoft Entra Connect is set up and finishes successfully.
Architecture patterns	Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none"> - Set up the Microsoft Entra Connect sync interval with the default or lowest value. Longer sync intervals increase the possibility of the Cloud PC provisioning failing in production due to a timeout. For more information, see Microsoft Entra hybrid join failing. - Set up Active Directory Domain Controller replication from a server in the same datacenter as the Windows 365 Azure network connection to provide faster replication. - Set up Microsoft Entra ID domain controller replication with a default value.

 [Expand table](#)

Dependency	Azure tenant readiness - Checks if Azure subscription is enabled, with no blocking restrictions, and is ready for use.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none"> - Use an account with the right privileges to manage the Azure, Intune, and Windows 365 subscriptions. For more information, see Role-based access control(RBAC). - Disable or modify any Azure policies that prevent the creation of Cloud PCs. For more information, see Restrict allowed VM SKUs. - Make sure the subscription has sufficient resource quotas for networking and general limits based on the maximum number of Cloud PCs to be created. Examples include the network gateway size, IP address space, size of the virtual network, and bandwidth required. For more information, see Networking limits and General limits.

 [Expand table](#)

Dependency	Azure virtual network readiness – Checks if the virtual network is in a supported Windows 365 region.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none"> - Create the virtual network in a Windows 365 Supported Azure regions for Cloud PC provisioning. - Create at least one subnet, in addition to the default subnet, to deploy the Cloud PC virtual network adaptors. - Where possible, create shared network services, such as Azure Firewall, VPN gateways, or ExpressRoute gateways, in a separate virtual network to allow for routing controls and expansion of deployment. <p>In virtual networks, apply network security groups (NSG) with appropriate exclusions to allow the required URLs for the Windows 365 service. For more information, see Networking requirements and Network security groups.</p>

[+] [Expand table](#)

Dependency	Azure subnet IP address usage – Checks if there are sufficient IP addresses available.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none"> - Create the virtual network with sufficient IP addresses to handle the Cloud PC creation and temporary IP address reservation during reprovision. It's recommended that you use an IP address space that's 1.5 to 2 times the maximum Cloud PCs that you deploy for the cloud. For more information, see General network requirements. - Treat the Azure virtual network as a logical extension of your on-premises network, and assign unique IP address space across all your networks to avoid routing conflicts.

[+] [Expand table](#)

Dependency	Endpoint connectivity – Checks if the external URLs needed for the Cloud PC provisioning are reachable from the virtual network.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none"> - Allow all the URLs needed for the Cloud PC provisioning via the Azure virtual network. For more information, see Allow network connectivity. - Use Azure Firewall to take advantage of Windows 365, Azure Virtual Desktop, and Intune FQDN tags to create application rules and allow URLs needed for the Windows 365 Cloud PC provisioning. For more information, see Use Azure Firewall to manage and secure Windows 365 environments. - Bypass or exclude Remote Desktop Protocol (RDP) traffic from any network inspection, proxying, or manipulation device to avoid latency and routing issues. For more information, see Traffic interception technologies. - From the end user device and network side, allow the Windows 365 service URLs and ports for proxy and network inspections. - Allow Azure internal IP addresses 168.63.129.16 and 169.254.169.254, as these IP addresses are used for communication with Azure platform services such as metadata or heartbeat. For more information, see What is IP address 168.63.129.16?, Azure Instance Metadata Service, and Virtual Network FAQ.

[] [Expand table](#)

Dependency	Intune enrollment – Checks if Intune allows Windows enrollment.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none"> - Ensure that Intune device type enrollment restrictions are set to allow the Windows mobile device management (MDM) platform for corporate enrollment. - For Microsoft Entra hybrid join, set up devices automatically by configuring the service connection point (SCP) for each domain in Microsoft Entra Connect or by using the targeted deployment model. For more information, see Configure Microsoft hybrid join and Microsoft Entra hybrid join targeted deployment.

[] [Expand table](#)

Dependency	First-party app permissions – Checks the Windows 365 app for permissions on the customer Azure subscription, resource group, and virtual network levels.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none"> - Ensure the account used for setting up the Azure network connection has read permissions on the Azure subscription in which the Azure virtual network is created. - Ensure in the Azure subscription that there are no policies in place that block permissions for the Windows 365 first-party app. The app must have permissions at the subscription, resource group, and virtual network level. <p>For more information, see Azure requirements.</p>

 [Expand table](#)

Dependency	Localization language pack – Checks if the language pack download locations are reachable.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none"> - Ensure the URLs needed for the appropriate version of Windows images are allowed via firewall rules used in the Azure virtual network. For more information, see Provide a localized Windows experience.

 [Expand table](#)

Dependency	RDP Shortpath – Checks if the User Datagram Protocol (UDP) configurations are in place for you to connect.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none"> - Enable RDP Shortpath for Cloud PC access to take advantage of UDP's resilience. For more information, see Use RDP Shortpath for public networks with Windows 365 and Use RDP Shortpath for private networks with Windows 365.

[+] [Expand table](#)

Dependency	Intune license – Checks if the tenant has appropriate Intune licenses to use Windows.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none">- Ensure Intune licenses are assigned to you in accordance with the licensing requirements.

[+] [Expand table](#)

Dependency	Single sign-on (SSO) check – Checks if the Kerberos server object is created in Active Directory and synced to Microsoft Entra ID.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none">- Ensure that the SSO option is selected in the provisioning policy. This option enables you to connect to the policy's Cloud PC by using sign in credentials from an Intune-managed physical device that's domain joined or Microsoft Entra joined. For more information, see Continue creating provisioning policies.

[+] [Expand table](#)

Dependency	DNS name resolution – Checks if the DNS in the Azure network connection can resolve on-premises Active Directory domain.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none">- Ensure the Azure virtual network is configured with the name resolution of an on-premises Microsoft Entra domain by using a custom DNS, a private DNS, or a private resolver. For more information, see What is Azure DNS?- Ensure the DNS servers configured in the virtual network are in the same geography and have the ability to register newly provisioned Cloud PCs

Dependency	DNS name resolution – Checks if the DNS in the Azure network connection can resolve on-premises Active Directory domain.
	without delays. Avoid DNS referral or redirections to prevent propagation delays, which can result in provisioning delays or failures.

[+] [Expand table](#)

Dependency	Microsoft Entra domain join – Checks that the credentials provided for Microsoft Entra domain join are valid and Cloud PCs can be domain joined.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none"> - Ensure the account provided for Microsoft Entra domain join has permissions on the Microsoft Entra organizational unit specified in the Azure network connection configuration. - Ensure the account provided isn't a standard user account with a domain join limitation. For more information, see Default limit to number of workstations a user can join to the domain. - Ensure the account specified is synced to Microsoft Entra ID. - Ensure the OU specified in the Azure network connection doesn't have any object limits. For more information, see Increase the computer account limit in the organizational unit.

For more information, see [Azure network connection health checks in Windows 365](#).

Azure network connection building blocks recommendations

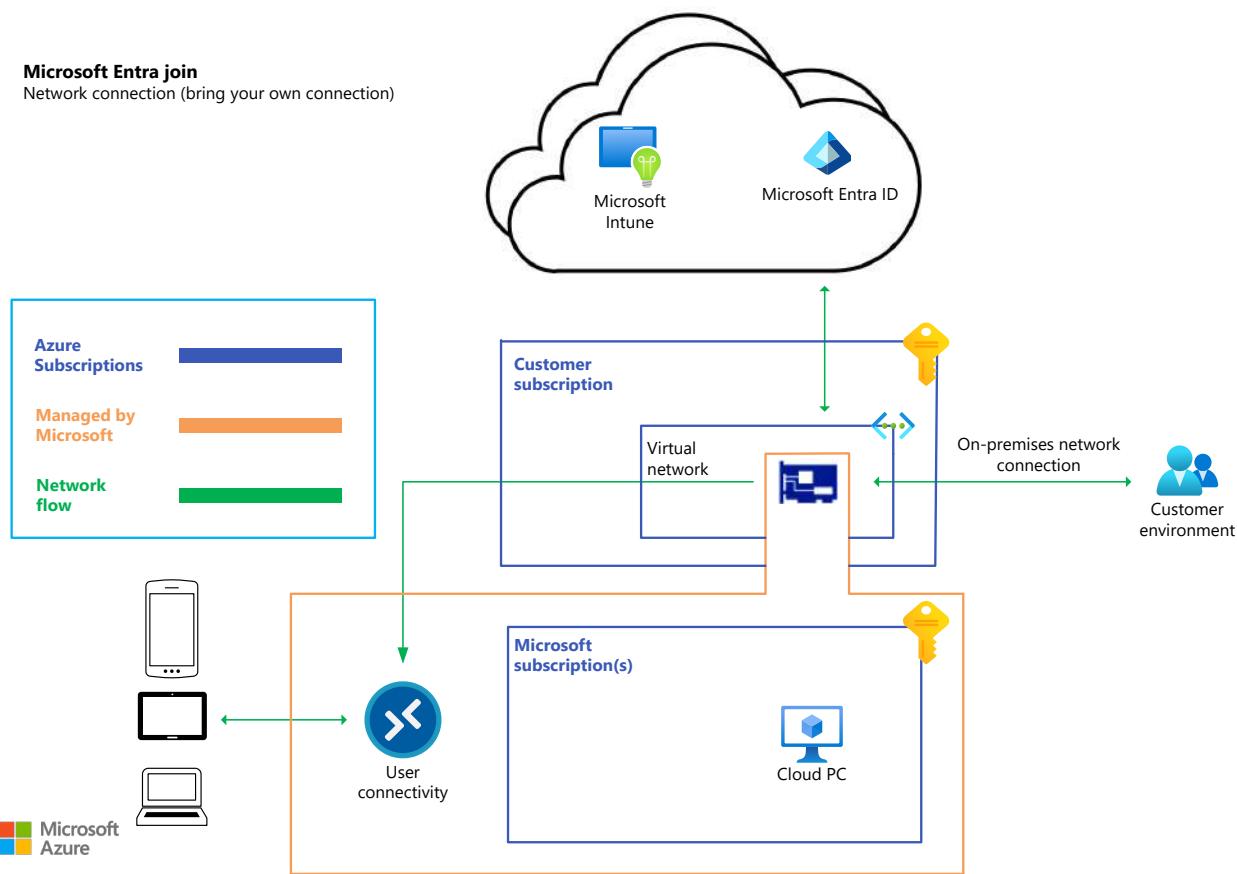
This section provides the breakdown of building blocks of the Windows 365 Azure network connection architecture pattern.

Azure subscription

Windows 365 usage in an Azure network connection architecture pattern involves two types of Azure subscriptions, a Microsoft subscription and a customer subscription.

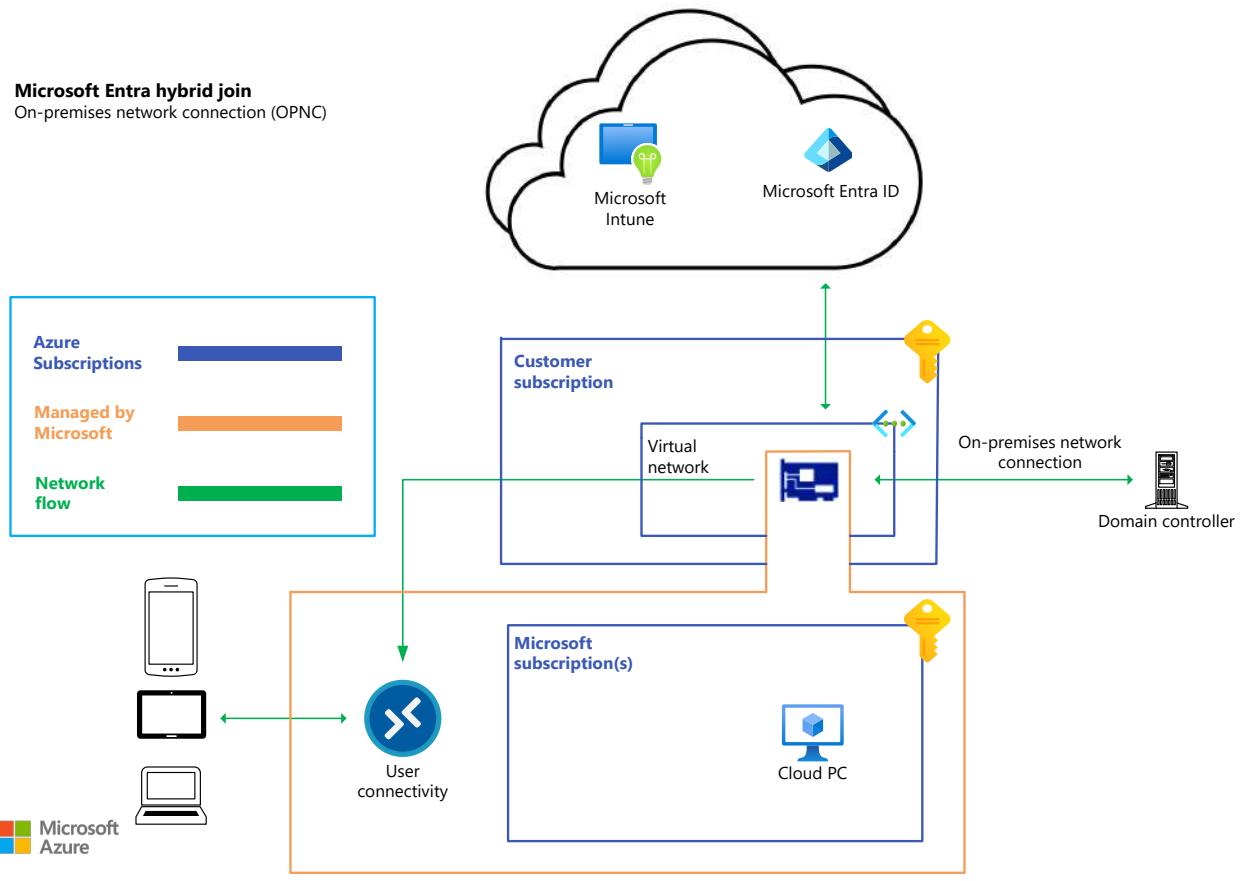
Windows 365 uses the *Hosted on behalf of* model to deliver services to Windows 365 customers. In this model, the Cloud PC is provisioned and run in Azure subscriptions owned by

Microsoft, while the network adapter of the Cloud PC is provisioned in a customer's Azure subscription. The following diagrams show two Azure network connection architecture patterns. Customers use their own Azure subscription and virtual network.



Download a [Visio file](#) of this architecture.

The previous architecture pattern uses the Microsoft Entra join identity to manage the Cloud PC.



Download a [Visio file](#) of this architecture.

The previous architecture pattern uses Microsoft Entra hybrid join identity to manage the Cloud PC and requires a *line of sight* network communication with Active Directory Domain Services (AD DS) domain controllers in on-premises environments.

[Expand table](#)

Component	Azure subscription – Azure subscription that hosts the virtual network used for providing connectivity for a Cloud PC to an on-premises environment and the internet.
Architecture patterns	Azure network connection for Microsoft Entra join, Azure network connection for Microsoft Entra hybrid join
Recommendations	<ul style="list-style-type: none"> - Create or use a subscription that has a virtual network and ExpressRoute or VPN gateways to provide a connection back to an on-premises environment. - Create a dedicated resource group for a Cloud PC to provide permission and resources management. - Exclude Cloud PC resource groups and virtual network from Azure policies that prevent automatic creation and deletion of virtual network interface card (vNIC) objects, and IP address assignment or release. For more

Component	Azure subscription – Azure subscription that hosts the virtual network used for providing connectivity for a Cloud PC to an on-premises environment and the internet.
	<p>information, see Lock your resources to protect your infrastructure and Azure requirements.</p> <ul style="list-style-type: none"> - Create dedicated virtual networks for better IP address management and routing controls.

Virtual Network and hybrid connection

Windows 365 Azure network connection-based architecture patterns require one or more Azure virtual networks. The virtual networks provide connectivity to on-premises environments and over the internet for provisioning a Cloud PC. The virtual network adapter of the Cloud PC is provisioned in the Azure virtual network of the customer-owned subscription as described in the [Azure subscription](#) section.

Azure networking can be deployed with varying design sophistication, based on the existing on-premises networking or Azure networking. To get started with a basic hybrid network design, see [Implement a secure hybrid network](#).

Consider the following factors when you design an Azure virtual network architecture:

- *IP address space*: The size of IP address space depends on the number of Cloud PCs to support. Plan for at least 1.5 times the maximum number of Cloud PCs that are deployed. The additional IP addresses account for IP addresses used during provisioning and deprovisioning of Cloud PCs.
- *Name resolution*: The DNS process used by the Cloud PC to resolve the on-premises domain name in a Microsoft Entra hybrid join deployment or to resolve internet resources or Azure resources in a Microsoft Entra join deployment model.
 - To use your existing on-premises DNS infrastructure, configure the IP addresses of one or more DNS servers for name resolution. For more information, see [DNS requirements](#).
 - Ensure the DNS server IP used in the Azure virtual network belong to the same geography as the Cloud PC and that it doesn't redirect DNS registration requests to another region. Otherwise, it results in delayed or failed deployments and Azure network connection health checks.
 - For Azure DNS-based name resolution, use the public or private Azure DNS or the private resolver option. For more information, see [Azure DNS documentation](#).
- *Network topology*: Azure networking supports topologies to accommodate different use cases.

- *Hub-spoke topology with virtual network peering*: This topology is the simplest way to provide an isolation of services with their own spoke and hub virtual networks. Shared services include Azure Firewall and network gateways. Choose this topology if you have a simple, single-site design to deploy a Cloud PC in one or more spoke virtual networks. For more information, see [Hub-and-spoke network topology](#).
- *Hub-spoke topology with Azure Virtual WAN*: Virtual WAN is an Azure networking service that brings together networking, security, and management capabilities that enable complex network requirements. Use this topology for multi-site, multi-region deployments with specific firewalling and routing requirements. For more information, see [Hub-spoke network topology with Virtual WAN](#).
- *Network gateway*: Azure network gateways provide connectivity from a virtual network to an on-premises network. There are VPN and ExpressRoute network gateways. Ensure that the maximum bandwidth requirements of a Cloud PC are considered before deciding on the ExpressRoute or VPN method of connectivity. Both VPN and ExpressRoute gateways are offered in tiers, or SKUs, that differ in the amount of bandwidth provided and other metrics. For more information, see [Extend an on-premises network using ExpressRoute](#) and [Connect an on-premises network to Azure using ExpressRoute](#).

Routing configurations

Windows 365 Azure network connection service uses automated health checks to determine the health and readiness of the customer's environment to provision Microsoft Entra join or Microsoft Entra hybrid join Cloud PCs in an Azure network connection-based architecture. Without proper routing configurations in your Azure virtual network and associated networking services, there's a high likelihood of failures or delays in your Cloud PC deployment. Consider the following recommendations to optimize routing for the Windows 365 network architecture:

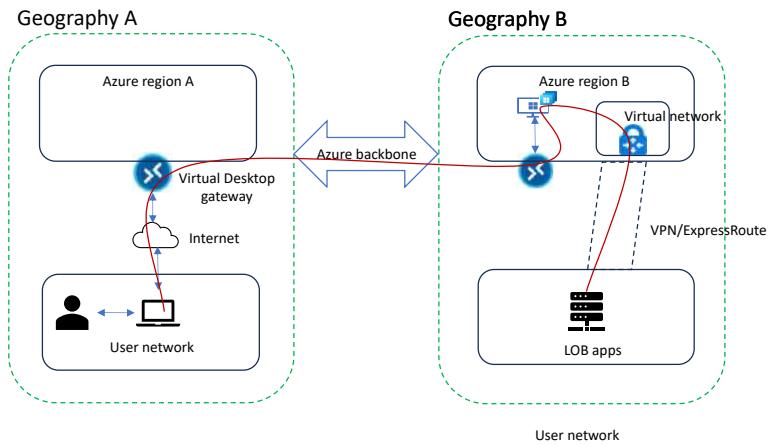
- *Allowlist required URLs*: Each Cloud PC deployed in Microsoft Entra hybrid join and Microsoft Entra join Azure network connection model requires several URLs to be allowed through OS anti-virus, network firewalls, and load balancers. Ensure all the URLs are allowed. For more information, see [Allow network connectivity](#).
- *Use Azure FQDN tags*: When you use the Azure Firewall service, use Azure FQDN tags to allow required URLs for Azure Virtual Desktop, Windows 365, and Intune. For more information, see [Use Azure Firewall to manage and secure Windows 365 environments](#).
- *Enable pass-through*: Windows 365 uses the RDP protocol, which is sensitive to latency introduced by traffic inspection devices such as a firewall or an SSL decryption appliance. Such latency can result in a poor experience, so disable traffic inspection of these URLs

and instead enable pass-through. For more information, see [Traffic interception technologies](#).

- *Bypass proxy*: Cloud and traditional proxy services, while suitable for internet access, introduce latency in RDP connections. This latency happens when the connection from the end user's physical device or from the Cloud PC is forced through a proxy and results in frequent disconnections, lags, and sluggish response times. Set `*.wvd.microsoft.com` and [Windows 365 gateway IP ranges](#) to bypass proxy services on the user's physical device, the network the physical device is connected to, and in the Cloud PC.

For more information, see [Optimizing RDP connectivity for Windows 365](#).

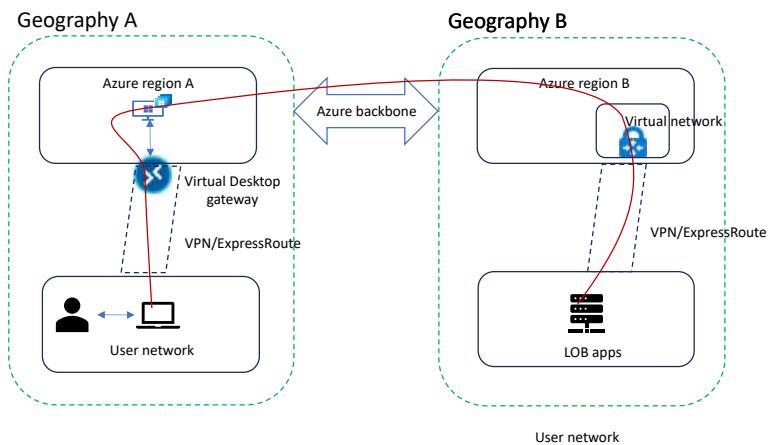
- *Shortest path routing*: Ensure RDP traffic from a Cloud PC reaches Virtual Desktop service endpoints via the shortest path. The ideal path is from a virtual network, directly to the Virtual Desktop gateway IP via the internet. Also ensure RDP traffic from the end user's physical device reaches the Virtual Desktop gateway IP directly. This configuration ensures optimal routing and doesn't degrade the user experience. Avoid routing RDP traffic to the internet via cloud proxy services or on-premises networks.
- *RDP Shortpath*: Enable RDP Shortpath-based access for end user networks, Azure networks, and Cloud PCs. RDP Shortpath uses UDP to transmit RDP traffic. Unlike TCP, it's resilient to high latency network connections. UDP also takes maximum advantage of the available network bandwidth to efficiently transfer RDP packets, which leads to an improved user experience. For more information, see [Use RDP Shortpath for public networks with Windows 365](#).
- *Cloud PC placement*: For an optimal user experience and routing performance, determine where customers are in relation to the work apps or network they access. Also consider the time customers spend accessing the LOB apps compared to the overall time that they access other apps. See the following two possible deployment options:
 - The following deployment model might be optimal if customers spend most of their work time accessing the LOB apps rather than work on locally installed apps, like apps in Microsoft 365. This model optimizes latency for LOB apps vs. Cloud PC access latency by placing the Cloud PC in the same region as the LOB app (Geography B). This optimization occurs even though the gateway is geographically closer to the end user (Geography A). The following diagram shows the possible traffic flow from the end user to the LOB apps.



Microsoft Azure

[Download a PowerPoint file](#) of this architecture.

- If customers occasionally access the LOB apps in Geography B, then deploying a Cloud PC closer to the customers might be optimal because it optimizes the Cloud PC access latency over LOB apps access latency. The following diagram shows how the traffic might flow in such a scenario.



Microsoft Azure

[Download a PowerPoint file](#) of this architecture.

AD DS recommendations

In a Microsoft Entra hybrid join architecture, an on-premises AD DS infrastructure acts as the identity source of authority. Having a properly configured and healthy AD DS infrastructure is a

crucial step to make the Windows 365 deployment successful.

On-premises AD DS supports many configurations and varying levels of complexity, so the recommendations provided only cover the baseline best practices.

- For Microsoft Entra hybrid join scenario, you can deploy AD DS in Azure VMs as described in the architecture reference in [Deploy AD DS in a virtual network](#). You can also use a hybrid network connection to provide a direct line of sight to your on-premises Microsoft Entra domain controller. For more information, see [Implement a secure hybrid network](#).
- For Microsoft Entra join deployment, follow the reference architecture in [Integrate on-premises Microsoft Entra domains with Microsoft Entra ID](#).
- Windows 365 uses a watchdog service as part of automated testing that creates a test VM account. That account shows as disabled in the organizational unit specified in Azure network connection configuration. Don't delete this account.
- Any Cloud PC that's decommissioned in the Microsoft Entra hybrid join model leaves behind a disabled computer account, which needs to be cleaned manually in AD DS.
- Microsoft Entra Domain Services isn't supported as an identity source because it doesn't support Microsoft Entra hybrid join.

DNS recommendations

In an Azure network connection deployment architecture, DNS servers or another DNS service used by an Azure virtual network is a crucial dependency. It's important to have a healthy infrastructure in place.

- For a Microsoft Entra hybrid join configuration, DNS should be able to resolve the domain to which the Cloud PC needs to be joined. There are multiple configuration options available, the simplest of them being specifying your DNS server IP in the Azure virtual network configuration. For more information, see [Name resolution that uses your own DNS server](#).
- Depending on the complexity of the infrastructure, such as a multi-region, multi-domain setup in Azure and on-premises environments, you should use a service like Azure DNS private zones or [Azure DNS Private Resolver](#).

Cloud PC connection recommendations

Deployed Cloud PCs should be configured to allow uninterrupted connection flow to and from the Virtual Desktop gateway service. Consider the following recommendations when you deploy apps as part of a Windows operating system configuration:

- Ensure that the VPS client doesn't launch when the user signs in because it can disconnect the session when the VPN tunnel establishes. The user would have to sign in a second time.
- Configure the VPN, proxy, firewall, and antivirus and antimalware apps to allow or bypass traffic bound for IP addresses 168.63.129.16 and 169.254.169.254. These IP addresses are used for communication with Azure platform services such as metadata and heartbeat. For more information, see [What is IP address 168.63.129.16?](#), [Azure Instance Metadata Service for virtual machines](#), and [Virtual Network FAQ](#).
- Don't manually modify the IP addresses of Cloud PCs because it might result in permanent disconnection. IP addresses are assigned with an indefinite lease and managed throughout the lifecycle of the Cloud PC by Azure networking services. For more information, see [Allocation methods](#).

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Ravishankar Nandagopalan](#) | Senior Product Manager

Other contributors:

- [Paul Collinge](#) | Principal Product Manager
- [Claus Emerich](#) | Principal Product Manager
- [David Falkus](#) | Principal Product Manager
- [Bob Roudebush](#) | Technical Leader and Cloud/Developer Technologist
- [Matt Shadbolt](#) | Principal Product Manager, Windows Cloud Experiences

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

[Plan your Cloud PC deployment](#)

[Windows 365 architecture](#)

[Windows 365 identity and authentication](#)

[Cloud PC lifecycle in Windows 365](#)

Related resources

Active Directory Domain Services overview

Data encryption in Windows 365

Understanding virtual desktop network connectivity

Web applications architecture design

Multiregion Business Continuity and Disaster Recovery (BCDR) for Azure Virtual Desktop

Azure Virtual Desktop

Azure Virtual Desktop is a comprehensive desktop and app virtualization service running on Microsoft Azure. Azure Virtual Desktop helps enable a secure remote desktop experience that helps organizations strengthen business resilience. It delivers simplified management, Windows 10 and 11 Enterprise multi-session, and optimizations for Microsoft 365 Apps for enterprise. With Azure Virtual Desktop, you can deploy and scale your Windows desktops and apps on Azure in minutes, providing integrated security and compliance features to help keep your apps and data secure.

As you continue to enable remote work for your organization with Azure Virtual Desktop, it's important to understand its disaster recovery (DR) capabilities and best practices. These practices strengthen reliability across regions to help keep data safe and employees productive. This article provides you with considerations on business continuity and disaster recovery (BCDR) prerequisites, deployment steps, and best practices. You learn about options, strategies, and architecture guidance. The content in this document enables you to prepare a successful BCDR plan and can help you bring more resilience to your business during planned and unplanned downtime events.

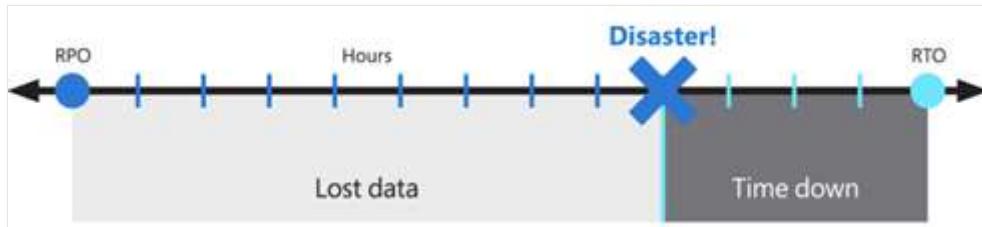
There are several types of disasters and outages, and each can have a different impact. Resiliency and recovery are discussed in depth for both local and region-wide events, including recovery of the service in a different remote Azure region. This type of recovery is called geo disaster recovery. It's critical to build your Azure Virtual Desktop architecture for resiliency and availability. You should provide maximum local resiliency to reduce the impact of failure events. This resiliency also reduces the requirements to execute recovery procedures. This article also provides information about high availability and best practices.

Goals and scope

The goals of this guide are to:

- Ensure maximum availability, resiliency, and geo-disaster recovery capability while minimizing data loss for important selected user data.
- Minimize recovery time.

These objectives are also known as the recovery point objective (RPO) and the Recovery Time Objective (RTO).



The proposed solution provides local high-availability, protection from a single [availability zone](#) failure, and protection from an entire Azure region failure. It relies on a redundant deployment in a different, or secondary, Azure region to recover the service. While it's still a good practice, Virtual Desktop and the technology used to build BCDR don't require Azure regions to be [paired](#). Primary and secondary locations can be any Azure region combination, if the network latency permits it. [Operating AVD host pools in multiple geographic regions](#) can offer more benefits not limited to BCDR.

To reduce the impact of a single availability zone failure, use resiliency to improve high availability:

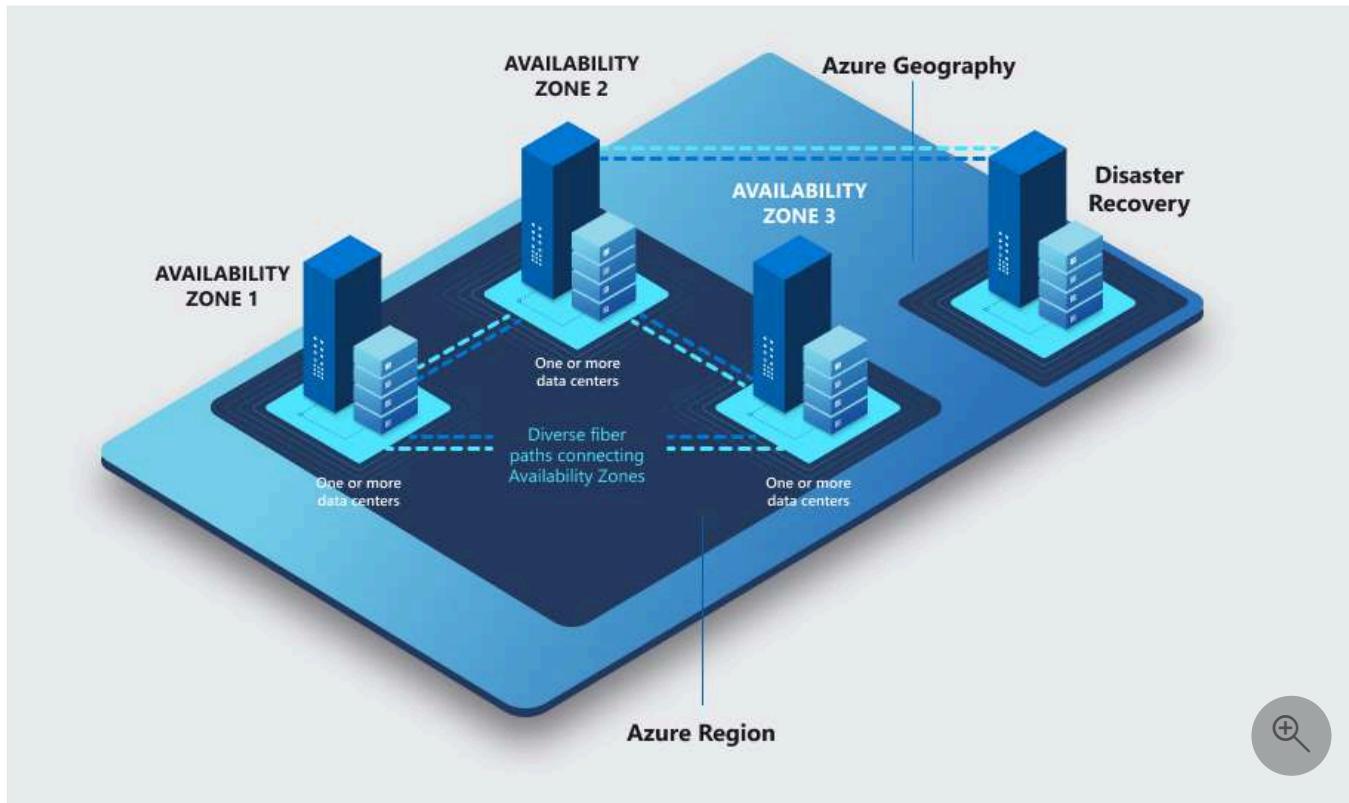
- At the [compute](#) layer, spread the Virtual Desktop session hosts across different availability zones.
- At the [storage](#) layer, use zone resiliency whenever possible.
- At the [networking](#) layer, deploy zone-resilient Azure ExpressRoute and virtual private network (VPN) gateways.
- For each dependency, review the impact of a single zone outage and plan mitigations. For example, deploy Active Directory Domain Controllers and other external resources accessed by Virtual Desktop users across multiple availability zones.

Depending on the number of availability zones you use, evaluate over-provisioning the number of session hosts to compensate for the loss of one zone. For example, even with $(n-1)$ zones available, you can ensure user experience and performance.

Note

Azure availability zones are a high-availability feature that can improve resiliency.

However, do not consider them a disaster recovery solution able to protect from region-wide disasters.



Because of the possible combinations of types, replication options, service capabilities, and availability restrictions in some regions, the [Cloud Cache](#) component from [FSLogix](#) is recommended to be used instead of storage-specific replication mechanisms.

Out of scope

OneDrive isn't covered in this article. For more information on redundancy and high-availability, see [SharePoint and OneDrive data resiliency in Microsoft 365](#).

Cost implications are discussed, but the primary goal is providing an effective geo disaster recovery deployment with minimal data loss. For more BCDR details, see the following resources:

- [BCDR considerations for Virtual Desktop](#)
- [Virtual Desktop disaster recovery](#)

Prerequisites

The design and implementation of Azure networking capabilities in Azure Virtual Desktop is critical for your Azure Virtual Desktop landing zone. Deploy the core infrastructure and make sure it's available in the primary and the secondary Azure region.

For detailed guidance on your network topology, you can use the Azure Cloud Adoption Framework [Network topology and connectivity](#) models:

- Traditional Azure networking topology
- Virtual WAN network topology (managed by Microsoft)
- Network topology and connectivity for Azure Virtual Desktop

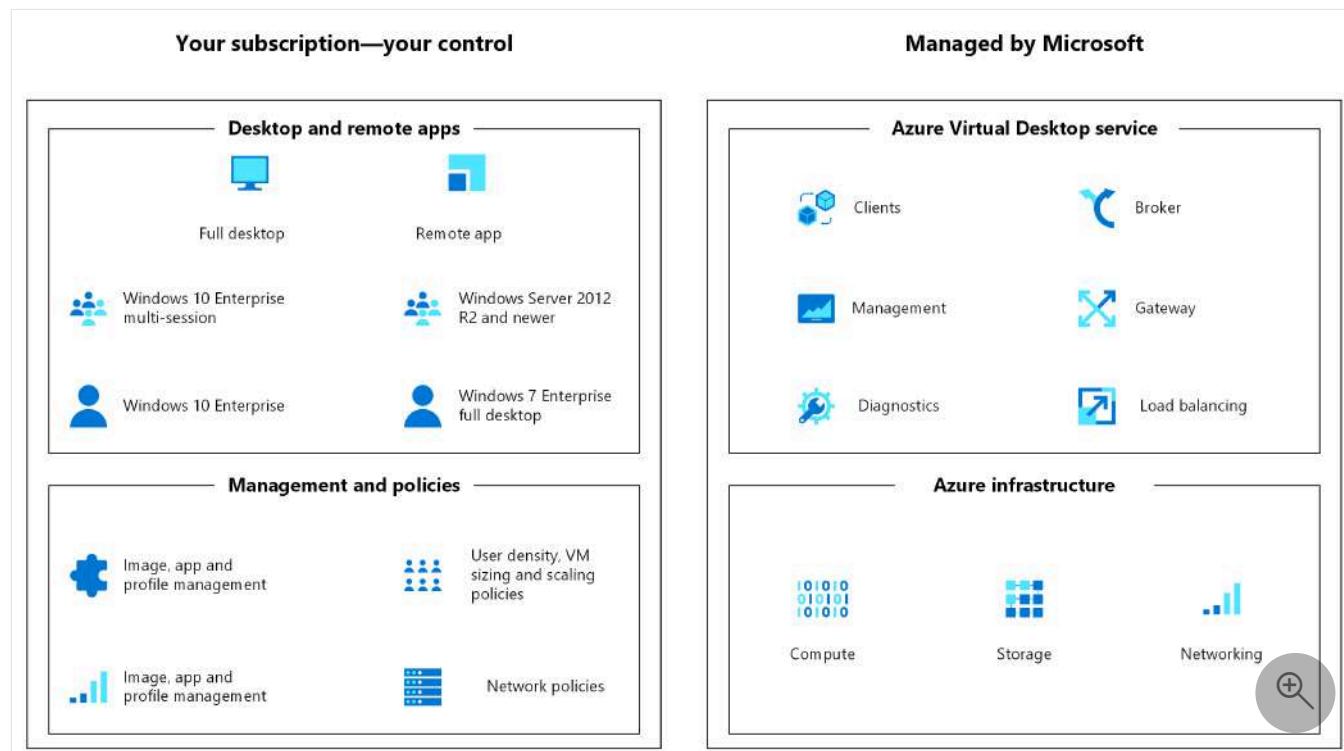
In both personal or pooled scenarios, deploy the primary Azure Virtual Desktop host pool and the secondary disaster recovery environment inside different spoke virtual networks and connect them to each hub in the same region. Place one hub in the primary location, one hub in the secondary location, and then establish connectivity between the two.

The hub eventually provides hybrid connectivity to on-premises resources, firewall services, identity resources like Active Directory Domain Controllers, and management resources like Log Analytics.

You should consider any line-of-business applications and dependent resource availability when failed over to the secondary location.

Azure Virtual Desktop control plane business continuity and disaster recovery

Azure Virtual Desktop is resilient to failures of individual components like Web service, Broker service, Gateway service, Resource directory, Geographical database and provides a reliable service to users. These components are Microsoft-managed as part of the Azure Virtual Desktop service and are located in around 40 Azure regions to be closer to users and to provide a resilient service.



When an outage occurs in a region, the service infrastructure components fail over to the secondary location and continue to be fully functioning. You can still access and configure service-related metadata, and users can still connect to available hosts. End-user connections stay online if the tenant environment or hosts remain accessible. [Data locations for Virtual Desktop](#) are different from the location of the host pool session host virtual machines (VMs) deployment. It's possible to locate Azure Virtual Desktop metadata in one of the supported regions and then deploy VMs in a different location. More details are provided in the [Azure Virtual Desktop service architecture and resilience](#) article.

The two different Azure Virtual Desktop host pool types support different recovery solutions.

- **Personal:** In this type of host pool, a user has a permanently assigned session host, which should never change. Since it's personal, this VM can store user data. The assumption is to use replication and backup techniques to preserve and protect the state.
- **Pooled:** Users are temporarily assigned one of the available session host VMs from the pool, either directly through a desktop application group or by using remote apps. VMs are stateless and user data and profiles are stored in external storage or OneDrive.

Active-Active vs. Active-Passive

If distinct sets of users have different BCDR requirements, Microsoft recommends that you use multiple host pools with different configurations. For example, users with a mission critical application might assign a fully redundant host pool with geo disaster recovery capabilities. However, development and test users can use a separate host pool with no disaster recovery at all.

For each single Virtual Desktop host pool, you can base your BCDR strategy on an active-active or active-passive model. This scenario assumes that the same set of users in one geographic location is served by a specific host pool.

- **Active-Active**
 - For each host pool in the primary region, you deploy a second host pool in the secondary region.
 - This configuration provides almost zero RTO, and RPO has an extra cost.
 - You don't require an administrator to intervene or fail over. During normal operations, the secondary host pool provides the user with Virtual Desktop resources.
 - Each host pool has its own storage accounts (at least one) for persistent user profiles.
 - You should evaluate latency based on the user's physical location and connectivity available. For some Azure regions, such as Western Europe and Northern Europe, the

difference can be negligible when accessing either the primary or secondary regions. You can validate this scenario using the [Azure Virtual Desktop Experience Estimator](#) tool.

- Users are assigned to different application groups, like *Desktop Application Group (DAG)* and *RemoteApp Application Group (RAG)*, in both the primary and secondary host pools. In this case, they see duplicate entries in their Virtual Desktop client feed. To avoid confusion, use separate Virtual Desktop workspaces with clear names and labels that reflect the purpose of each resource. Inform your users about the usage of these resources.



- If you need storage to manage [FSLogix Profile](#) and [ODFC containers](#) separately, use Cloud Cache to ensure almost zero RPO.
- To avoid profile conflicts, don't permit users to access both host pools at the same time.
- Due to the active-active nature of this scenario, you should educate your users on how to use these resources in the proper way.

Note

Using separate [ODFC containers](#) is an advanced scenario with higher complexity. Deploying this way is recommended only in some [specific scenarios](#).

- **Active-Passive**
 - Like active-active, for each host pool in the primary region, you deploy a second host pool in the secondary region.
 - The amount of compute resources active in the secondary region is reduced compared to the primary region, depending on the budget available. You can use automatic

scaling to provide more compute capacity, but it requires more time, and Azure capacity isn't guaranteed.

- This configuration provides higher RTO when compared to the active-active approach, but it's less expensive.
- You need administrator intervention to execute a failover procedure if there's an Azure outage. The secondary host pool doesn't normally provide the user access to Virtual Desktop resources.
- Each host pool has its own storage accounts for persistent user profiles.
- Users that consume Virtual Desktop services with optimal latency and performance are affected only if there's an Azure outage. You should validate this scenario by using the [Azure Virtual Desktop Experience Estimator](#) tool. Performance should be acceptable, even if degraded, for the secondary disaster recovery environment.
- Users are assigned to only one set of application groups, like Desktop and Remote apps. During normal operations, these apps are in the primary host pool. During an outage, and after a failover, users are assigned to Application Groups in the secondary host pool. No duplicate entries are shown in the user's Virtual Desktop client feed, they can use the same workspace, and everything is transparent for them.
- If you need storage to manage FSLogix Profile and Office containers, use Cloud Cache to ensure almost zero RPO.
 - To avoid profile conflicts, don't permit users to access both host pools at the same time. Since this scenario is active-passive, administrators can enforce this behavior at the application group level. Only after a failover procedure is the user able to access each application group in the secondary host pool. Access is revoked in the primary host pool application group and reassigned to an application group in the secondary host pool.
 - Execute a failover for all application groups, otherwise users using different application groups in different host pools might cause profile conflicts if not effectively managed.
- It's possible to allow a specific subset of users to selectively fail over to the secondary host pool and provide limited active-active behavior and test failover capability. It's also possible to fail over specific application groups, but you should educate your users to not use resources from different host pools at the same time.

For specific circumstances, you can create a single host pool with a mix of session hosts located in different regions. The advantage of this solution is that if you have a single host pool, then there's no need to duplicate definitions and assignments for desktop and remote apps.

Unfortunately, [disaster recovery for shared host pools](#) has several disadvantages:

- For pooled host pools, it isn't possible to force a user to a session host in the same region.

- A user might experience higher latency and suboptimal performance when connecting to a session host in a remote region.
- If you require storage for user profiles, you need a complex configuration to manage assignments for session hosts in the primary and secondary regions.
- You can use drain mode to temporarily disable access to session hosts located in the secondary region. But this method introduces more complexity, management overhead, and inefficient use of resources.
- You can maintain session hosts in an offline state in the secondary regions, but it introduces more complexity and management overhead.

Considerations and recommendations

General

In order to deploy either an active-active or active-passive configuration using multiple host pools and an FSLogix cloud cache mechanism, you can create the host pool inside the same workspace or a different one, depending on the model. This approach requires you to maintain the alignment and updates, keeping both host pools in sync and at the same configuration level. In addition to a new host pool for the secondary disaster recovery region, you need:

- To create new distinct application groups and related applications for the new host pool.
- To revoke user assignments to the primary host pool, and then manually reassign them to the new host pool during the failover.

Review the [Business continuity and disaster recovery options for FSLogix](#).

- [No profile recovery](#) isn't covered in this document.
- [Cloud cache \(active/passive\)](#) is included in this document but is implemented it using the same host pool.
- [Cloud cache \(active/active\)](#) is covered in the remaining part of this document.

There are limits for Virtual Desktop resources that must be considered in the design of a Virtual Desktop architecture. Validate your design based on the [Virtual Desktop service limits](#).

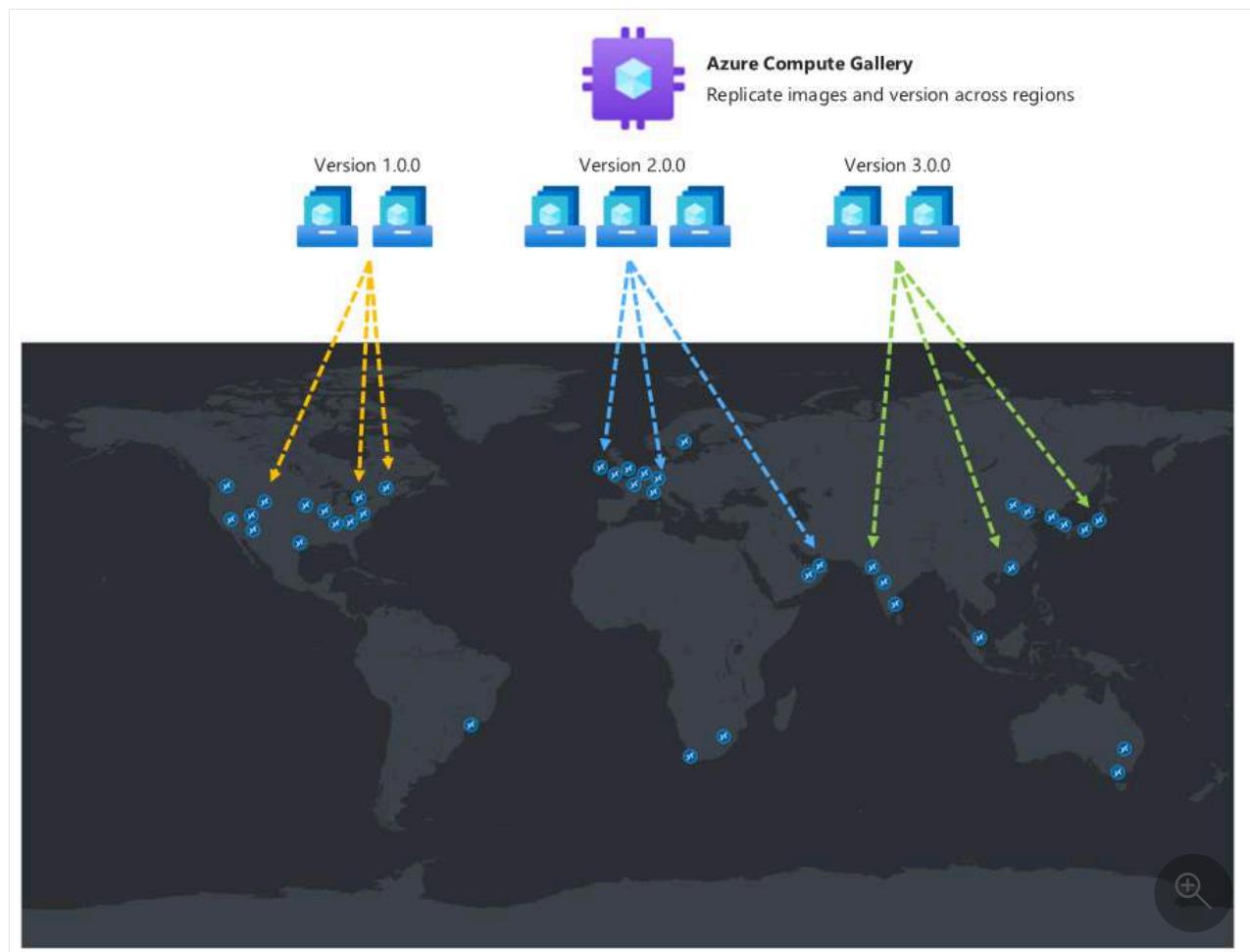
For diagnostics and monitoring, it's good practice to use the same Log Analytics workspace for both the primary and secondary host pool. Using this configuration, [Azure Virtual Desktop Insights](#) offers a unified view of deployment in both regions.

However, using a single log destination can cause problems if the entire primary region is unavailable. The secondary region won't be able to use the Log Analytics workspace in the unavailable region. If this situation is unacceptable, the following solutions could be adopted:

- Use a separate Log Analytics workspace for each region, and then point the Virtual Desktop components to log toward its local workspace.
- Test and review Logs Analytics [workspace replication and failover capabilities](#).

Compute

- For the deployment of both host pools in the primary and secondary disaster recovery regions, you should spread your session host VM fleet across multiple availability zones. If availability zones aren't available in the local region, you can use an availability set to make your solution more resilient than with a default deployment.
- The golden image that you use for host pool deployment in the secondary disaster recovery region should be the same you use for the primary. You should store images in the Azure Compute Gallery and configure multiple image replicas in both the primary and the secondary locations. Each image replica can sustain a parallel deployment of a maximum number of VMs, and you might require more than one based on your desired deployment batch size. For more information, see [Store and share images in an Azure Compute Gallery](#).



- The Azure Compute Gallery isn't a global resource. It's recommended to have at least a secondary gallery in the secondary region. In your primary region, create a gallery, a VM

image definition and a VM image version. Then, create the same objects also in the secondary region. When creating the VM image version, there's the possibility to copy the VM image version created in the primary region by specifying the gallery, VM image definition and VM image version used in the primary region. Azure copies the image and creates a local VM image version. It's possible to execute this operation using the Azure portal or the Azure CLI command as outlined below:

[Create an image definition and an image version](#)

`az sig image-version create`

- Not all the session host VMs in the secondary disaster recovery locations must be active and running all the time. You must initially create a sufficient number of VMs, and after that, use an autoscale mechanism like [Scaling plans](#). With these mechanisms, it's possible to maintain most compute resources in an offline or deallocated state to reduce costs.
- It's also possible to use automation to create session hosts in the secondary region only when needed. This method optimizes costs, but depending on the mechanism you use, might require a longer RTO. This approach doesn't permit failover tests without a new deployment and doesn't permit selective failover for specific groups of users.

! Note

You must power on each session host VM for a few hours at least one time every 90 days to refresh the authentication token needed to connect to the Virtual Desktop control plane. You should also routinely apply security patches and application updates.

- Having session hosts in an offline, or *deallocated*, state in the secondary region doesn't guarantee that capacity is available in case of a primary region-wide disaster. It also applies if new session hosts are deployed on-demand when needed, and with [Site Recovery](#) usage. Compute capacity can be guaranteed only if the related resources are already allocated and active.

i Important

[Azure Reservations](#) doesn't provide guaranteed capacity in the region.

For Cloud Cache usage scenarios, we recommend using the Premium tier for managed disks.

Storage

In this guide, you use at least two separate storage accounts for each Virtual Desktop host pool. One is for the FSLogix Profile container, and one is for the Office container data. You also need one more storage account for [MSIX](#) packages. The following considerations apply:

- You can use [Azure Files](#) share and [Azure NetApp Files](#) as storage alternatives. To compare the options, see the [FSLogix container storage options](#).
- Azure Files share can provide zone resiliency by using the zone-redundant storage (ZRS) resiliency option, if it's available in the region.
- You can't use the geo-redundant storage feature in the following situations:
 - You require a [region that doesn't have a pair](#). The region pairs for geo-redundant storage are fixed and can't be changed.
 - You're using the Premium tier.
- RPO and RTO are higher compared to FSLogix Cloud Cache mechanism.
- It isn't easy to test failover and failback in a production environment.
- Azure NetApp Files requires more considerations:
 - Zone redundancy isn't yet available. If the resiliency requirement is more important than performance, use Azure Files share.
 - Azure NetApp Files can be [zonal](#), that is customers can decide in which (single) Azure Availability Zone to allocate.
 - [Cross-zone replication](#) can be established at the volume level to provide zone resiliency but replication happens asynchronous and requires manual failover. This process requires a recovery point objective (RPO) and recovery time objective (RTO) that are greater than zero. Before using this feature, review the [requirements and considerations for cross-zone replication](#).
 - You can use Azure NetApp Files with zone-redundant VPN and ExpressRoute gateways, if [standard networking](#) feature is used, which you might use for networking resiliency. For more information, see [Supported network topologies](#).
 - Azure Virtual WAN is supported when used together with Azure NetApp Files [standard networking](#). For more information, see [Supported network topologies](#).
- Azure NetApp Files has a [cross-region replication mechanism](#). The following considerations apply:
 - It's not available in all regions.
 - [Cross-region replication of Azure NetApp Files volumes](#) region pairs can be different than Azure Storage region pairs.
 - It can't be used at the same time with [cross-zone replication](#)
 - Failover isn't transparent, and failback requires [storage reconfiguration](#).
- Limits
 - There are limits in the size, input/output operations per second (IOPS), bandwidth MBps for both [Azure Files share](#) and [Azure NetApp Files](#) storage accounts and volumes. If necessary, it's possible to use more than one for the same host pool in

Virtual Desktop by using [per-group settings](#) in FSLogix. However, this configuration requires more planning and configuration.

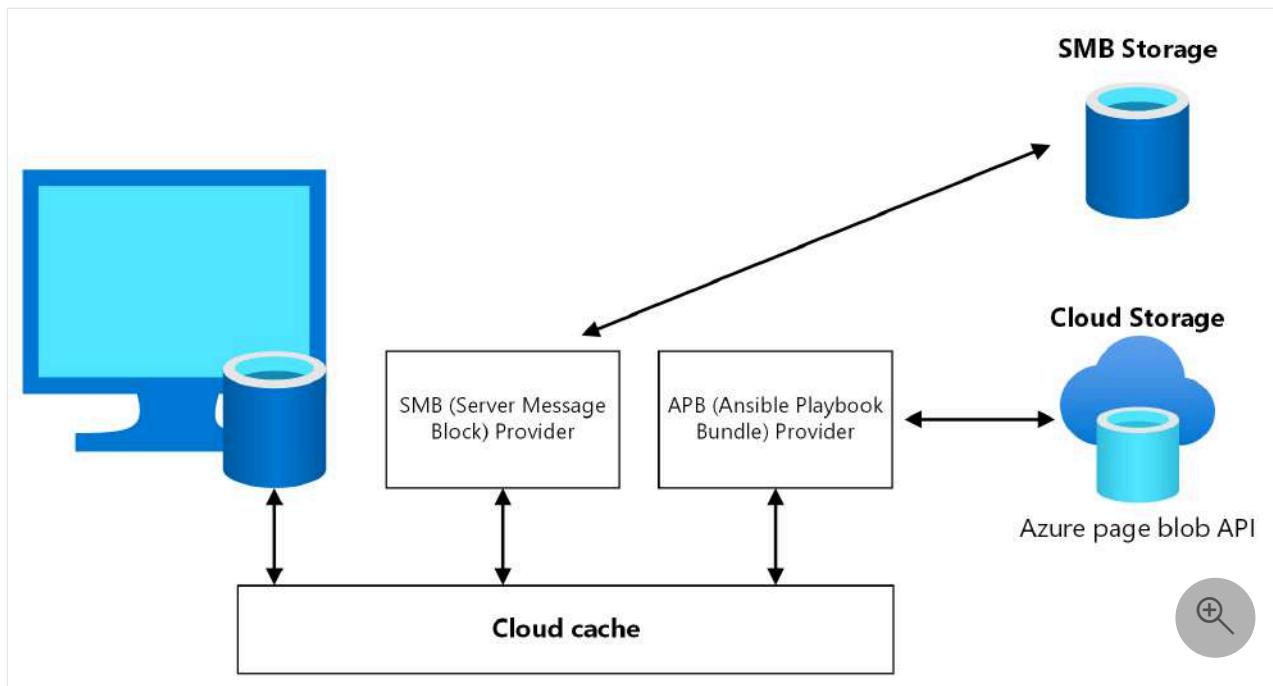
The storage account you use for MSIX application packages should be distinct from the other accounts for Profile and Office containers. The following Geo-disaster recovery options are available:

- **One storage account with geo-redundant storage enabled, in the primary region**
 - The secondary region is fixed. This option isn't suitable for local access if there's storage account failover.
- **Two separate storage accounts, one in the primary region and one in the secondary region (recommended)**
 - Use zone-redundant storage for at least the primary region.
 - Each host pool in each region has local storage access to MSIX packages with low latency.
 - Copy MSIX packages twice in both locations and register the packages twice in both host pools. Assign users to the application groups twice.

FSLogix

Microsoft recommends that you use the following FSLogix configuration and features:

- If the Profile container content needs to have separate BCDR management, and has different requirements compared to the Office container, you should split them.
 - Office Container only has cached content that can be rebuilt or repopulated from the source if there's a disaster. With Office Container, you might not need to keep backups, which can reduce costs.
 - When using different storage accounts, you can only enable backups on the profile container. Or, you must have different settings like retention period, storage used, frequency, and RTO/RPO.
- [Cloud Cache](#) is an FSLogix component in which you can specify multiple profile storage locations and asynchronously replicate profile data, all without relying on any underlying storage replication mechanisms. If the first storage location fails or isn't reachable, Cloud Cache automatically fails over to use the secondary, and effectively adds a resiliency layer. Use Cloud Cache to replicate both Profile and Office containers between different storage accounts in the primary and secondary regions.



- You must enable Cloud Cache twice in the session host VM registry, once for [Profile Container](#) and once for [Office Container](#). It's possible to not enable Cloud Cache for Office Container, but not enabling it might cause a data misalignment between the primary and the secondary disaster recovery region if there's failover and failback. Test this scenario carefully before using it in production.
- Cloud Cache is compatible with both [profile split](#) and [per-group](#) settings. per-group requires careful design and planning of active directory groups and membership. You must ensure that every user is part of exactly one group, and that group is used to grant access to host pools.
- The *CCDLocations* parameter specified in the registry for the host pool in the secondary disaster recovery region is reverted in order, compared to the settings in the primary region. For more information, see [Tutorial: Configure Cloud Cache to redirect profile containers or office container to multiple Providers](#).

💡 Tip

This article focuses on a specific scenario. Additional scenarios are described in [High availability options for FSLogix](#) and [Business continuity and disaster recovery options for FSLogix](#).

The following example shows a Cloud Cache configuration and related registry keys:

Primary Region = North Europe

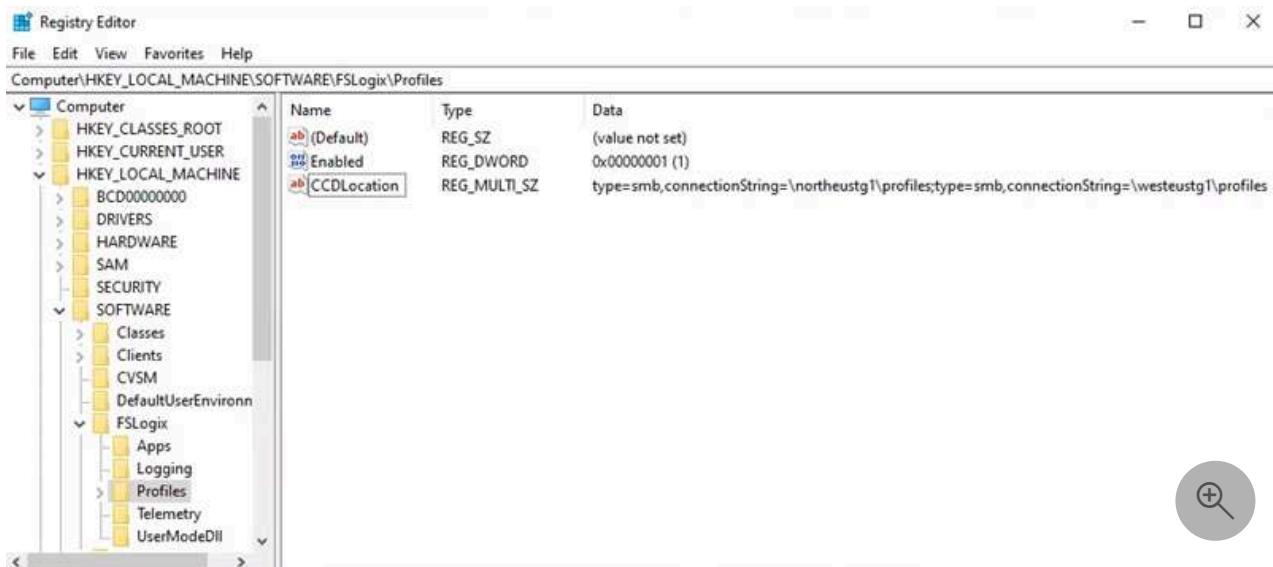
- Profile container storage account URI = \\northeustg1\\profiles
 - Registry Key path = **HKEY_LOCAL_MACHINE > SOFTWARE > FSLogix > Profiles**

- *CCDLocations* value =

type=smb,connectionString=\northeustg1\profiles;type=smb,connectionString=\westeustg1\profiles

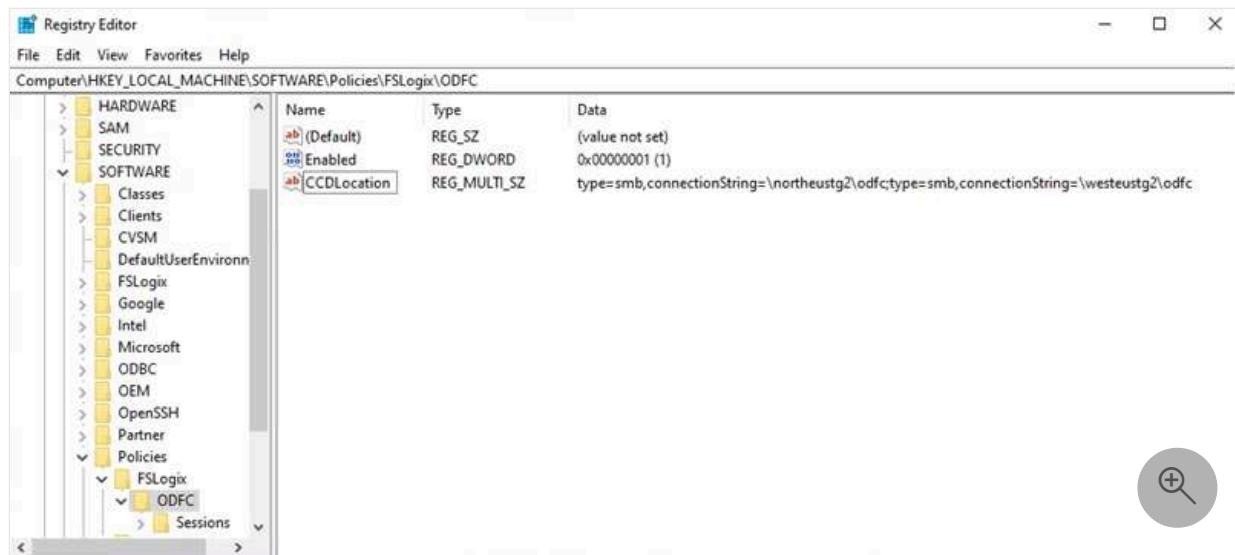
! Note

If you previously downloaded the **FSLogix Templates**, you can accomplish the same configurations through the Active Directory Group Policy Management Console. For more information about how to set up the Group Policy Object for FSLogix, see [Use FSLogix Group Policy Template Files](#).



- Office container storage account URI = \northeustg2\odfc
 - Registry Key path = **HKEY_LOCAL_MACHINE > SOFTWARE > Policy > FSLogix > ODFC**
 - *CCDLocations* value =

type=smb,connectionString=\northeustg2\odfc;type=smb,connectionString=\westeustg2\odfc



! Note

In the screenshots above, not all the recommended registry keys for FSLogix and Cloud Cache are reported, for brevity and simplicity. For more information, see [FSLogix configuration examples](#).

Secondary Region = West Europe

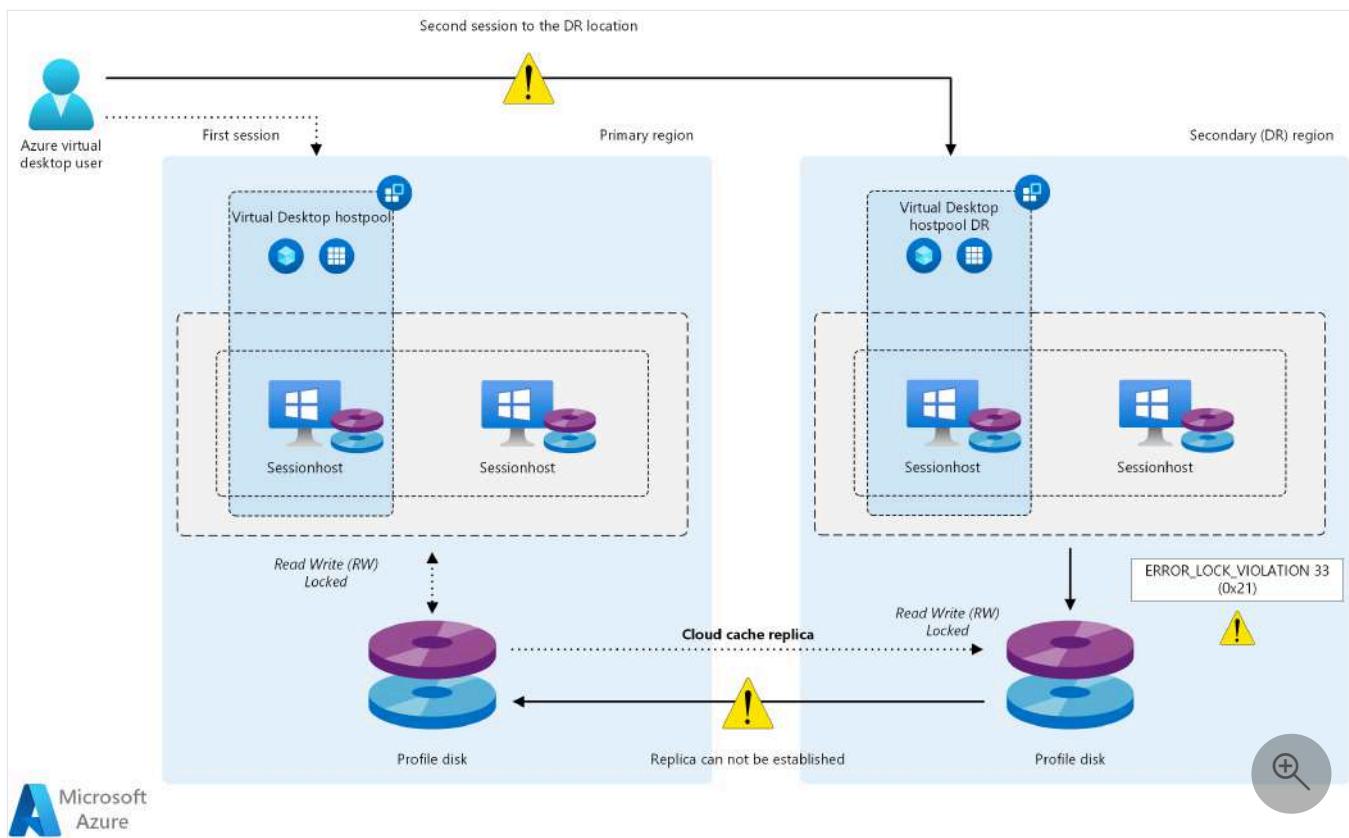
- Profile container storage account URI = \westeustg1\profiles
 - Registry Key path = **HKEY_LOCAL_MACHINE > SOFTWARE > FSLogix > Profiles**
 - CCDLocations value =

type=smb,connectionString=\westeustg1\profiles;type=smb,connectionString=\northeastg1\profiles
- Office container storage account URI = \westeustg2\odfc
 - Registry Key path = **HKEY_LOCAL_MACHINE > SOFTWARE > Policy > FSLogix > ODPC**
 - CCDLocations value =

type=smb,connectionString=\westeustg2\odfc;type=smb,connectionString=\northeastg2\odfc

Cloud Cache replication

The Cloud Cache configuration and replication mechanisms guarantee profile data replication between different regions with minimal data loss. Since the same user profile file can be opened in ReadWrite mode by only one process, concurrent access should be avoided, thus users shouldn't open a connection to both host pools at the same time.



Download a [Visio file](#) of this architecture.

Dataflow

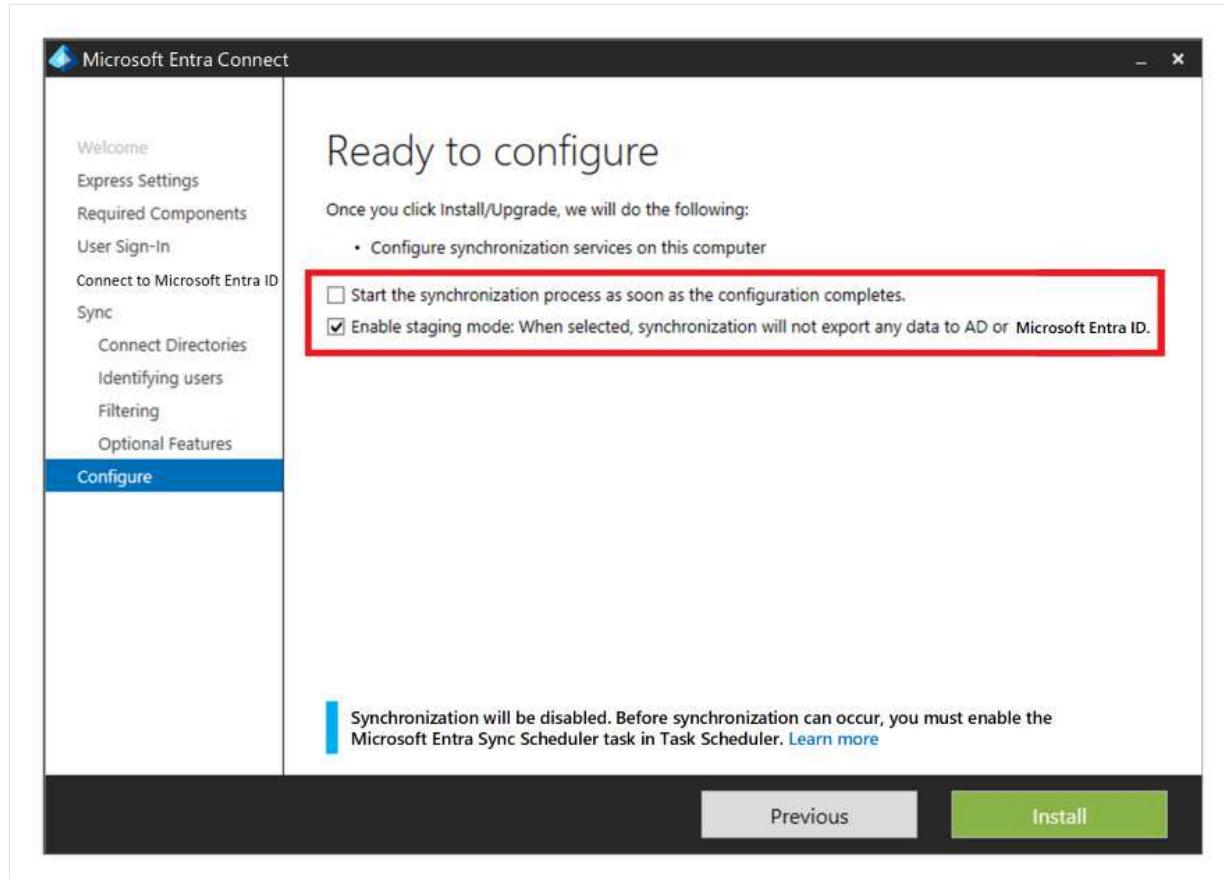
1. An Azure Virtual Desktop user launches Virtual Desktop client, and then opens a published Desktop or Remote App application assigned to the primary region host pool.
2. FSLogix retrieves the user Profile and Office containers, and then mounts the underlying storage VHD/X from the storage account located in the primary region.
3. At the same time, the Cloud Cache component initializes replication between the files in the primary region and the files in the secondary region. For this process, Cloud Cache in the primary region acquires an exclusive read-write lock on these files.
4. The same Virtual Desktop user now wants to launch another published application assigned on the secondary region host pool.
5. The FSLogix component running on the Virtual Desktop session host in the secondary region tries to mount the user profile VHD/X files from the local storage account. But the mounting fails since these files are locked by the Cloud Cache component running on the Virtual Desktop session host in the primary region.
6. In the default FSLogix and Cloud Cache configuration, the user can't sign in and an error is tracked in the FSLogix diagnostic logs, *ERROR_LOCK_VIOLATION 33 (0x21)*.

Operational events:				
Type	Id	Date	Description	
>Error	41	4/1/2022 20:38:05	Operation: PSLogon.Logon_OOFC, SessionId: 2, ErrorCode: 33, Detail: Logon failed, Please check logs and tracing and verify that the users disk was detached.	
>Error	26	4/1/2022 20:38:05	Error (The process cannot access the file because another process has locked a portion of the file.)	
>Error	26	4/1/2022 20:38:05	Frx.Service.OOFContainer.cpp(483): [WCODE: 0x00000021] Base disk failed to acquire an exclusive lock (The process cannot access the file because another process has locked a portion of the file.)	
>Error	41	4/1/2022 20:38:02	Operation: PSLogon.Logon_PROFILE, SessionId: 2, ErrorCode: 33, Detail: Logon failed, Please check logs and tracing and verify that the users disk was detached.	
>Error	26	4/1/2022 20:37:57	LoadProfile failed. Version: 2.5.7979.62179 User: AdminDev. SID: S-1-12-1-000000000-111111111-222222222-333333333. SessionId: 2 (The process cannot access the file because another process has locked a portion of the file.)	

Identity

One of the most important dependencies for Azure Virtual Desktop is the availability of user identity. To access full remote virtual desktops and remote apps from your session hosts, your users need to be able to authenticate. [Microsoft Entra ID](#) is Microsoft's centralized cloud identity service that enables this capability. Microsoft Entra ID is always used to authenticate users for Virtual Desktop. Session hosts can be joined to the same Microsoft Entra tenant, or to an Active Directory domain using [Active Directory Domain Services \(AD DS\)](#) or Microsoft Entra Domain Services, providing you with a choice of flexible configuration options.

- **Microsoft Entra ID**
 - It's a global multi-region and resilient service with [high-availability](#). No other action is required in this context as part of a Virtual Desktop BCDR plan.
- **Active Directory Domain Services**
 - For Active Directory Domain Services to be resilient and highly available, even if there's a region-wide disaster, you should deploy at least two domain controllers (DCs) in the primary Azure region. These domain controllers should be in different availability zones if possible, and you should ensure proper replication with the infrastructure in the secondary region and eventually on-premises. You should create at least one more domain controller in the secondary region with global catalog and DNS roles. For more information, see [Deploy Active Directory Domain Services \(AD DS\) in an Azure virtual network](#).
- **Microsoft Entra Connect**
 - If you're using Microsoft Entra ID with Active Directory Domain Services, and then [Microsoft Entra Connect](#) to synchronize user identity data between Active Directory Domain Services and Microsoft Entra ID, you should consider the resiliency and recovery of this service for protection from a permanent disaster.
 - You can provide high availability and disaster recovery by installing a second instance of the service in the secondary region and enable [staging mode](#).
 - If there's a recovery, the administrator is required to promote the secondary instance by taking it out of staging mode. They must follow the procedure to [switch the active server](#) as at least a [Hybrid Identity Administrator](#).

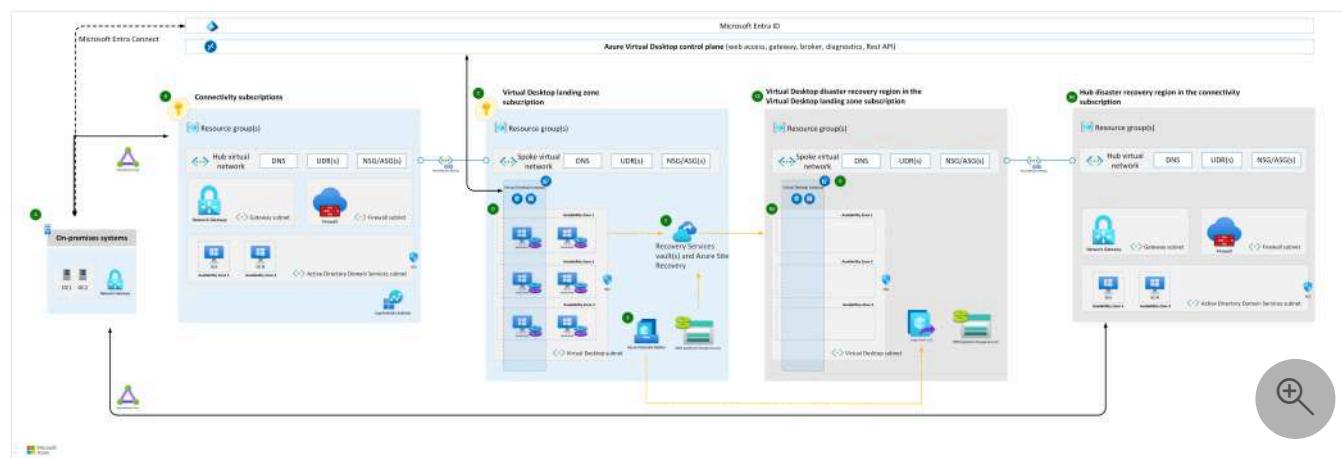


- **Microsoft Entra Domain Services**

- You can use Microsoft Entra Domain Services in some scenarios as an alternative to Active Directory Domain Services.
- It offers [high-availability](#).
- If geo-disaster recovery is in scope for your scenario, you should deploy another replica in the secondary Azure region by using a [replica set](#). You can also use this feature to increase high availability in the primary region.

Architecture diagrams

Personal host pool

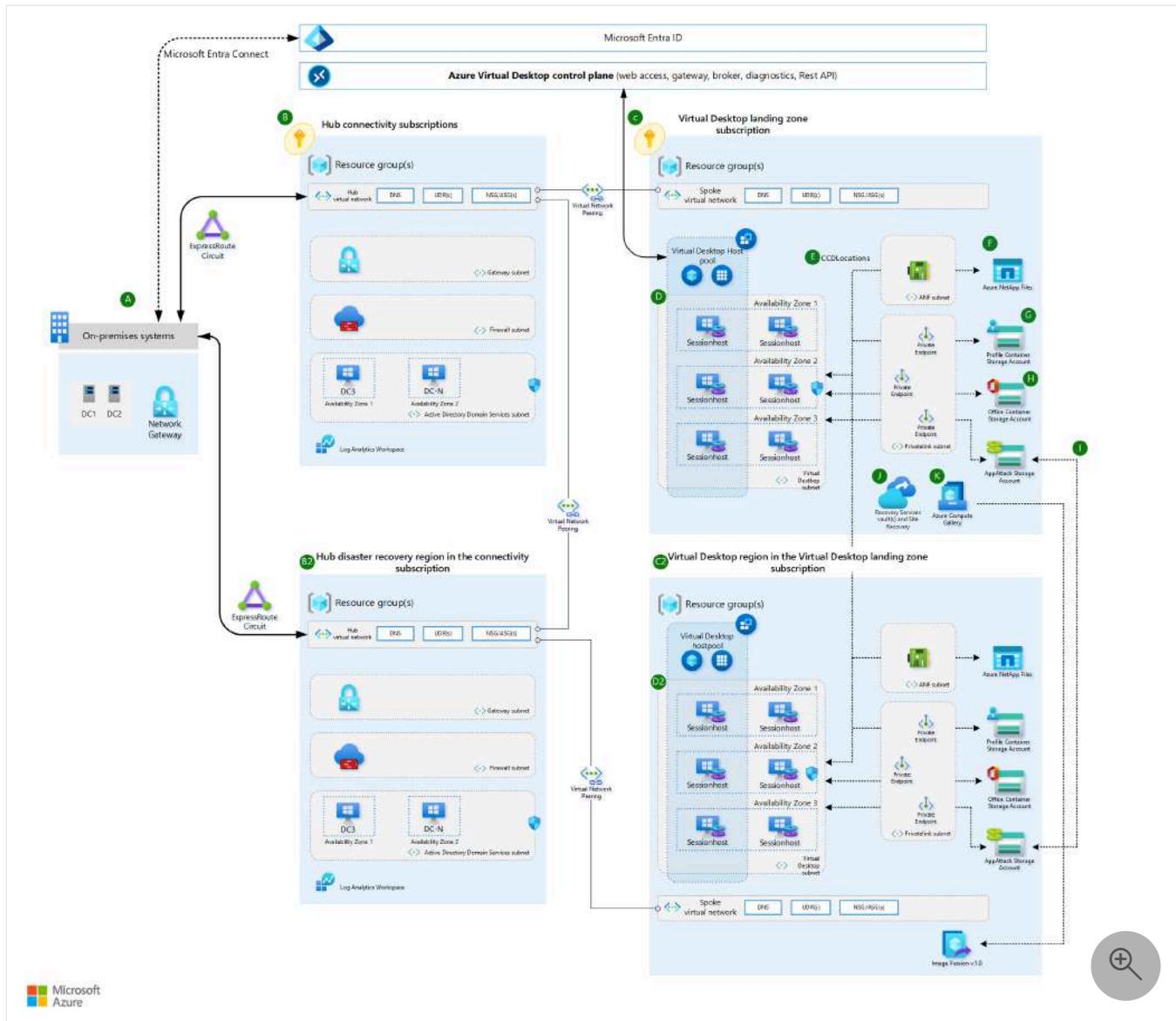


Download a [Visio file](#) of this architecture.

 Expand table

Design area	Description
A	One of the most important dependencies for Azure Virtual Desktop is the availability of user identity. To access full remote virtual desktops and remote apps from your session hosts, your users need to be able to authenticate. Review the Identity option above.
B	If Azure Virtual Desktop users need access to on-premises resources, it's critical that you consider high availability in the network infrastructure that's required to connect to the resources. Assess and evaluate the resiliency of your authentication infrastructure, and consider BCDR aspects for dependent applications and other resources. These considerations will help ensure availability in the secondary disaster recovery location.
C	Depending on the size of your deployment and organization structure ensure all subscription have enough quota to run Azure Virtual Desktop workloads in different regions and that you have the right Azure role-based access control (Azure RBAC) roles assigned.
D	For the deployment of both host pools in the primary and secondary disaster recovery regions, you should spread your session host VM fleet across multiple availability zones. If availability zones aren't available in the local region, you can use an availability set to make your solution more resilient than with a default deployment.
E	The golden image that you use for host pool deployment in the secondary disaster recovery region should be the same you use for the primary. You should store images in the Azure Compute Gallery and configure multiple image replicas in both the primary and the secondary locations.
F	You can use Azure Site Recovery or a secondary host pool (hot standby) to maintain a backup environment.
G	You can create a new host pool in the failover region and keep all the resources turned off. For this method, set up new application groups in the failover region and assign users to the groups. Then, you can use a recovery plan in Site Recovery to turn on host pools and create an orchestrated process.

Pooled host pool



Download a [Visio file](#) of this architecture.

[Expand table](#)

Design area	Description
A	<p>One of the most important dependencies for Azure Virtual Desktop is the availability of user identity. To access full Azure Virtual Desktops and remote apps from your session hosts, your users need to be able to authenticate. Review the Identity option above.</p>
B	<p>If Azure Virtual Desktop users need access to on-premises resources, it's critical that you consider high availability in the network infrastructure that's required to connect to the resources. Assess and evaluate the resiliency of your authentication infrastructure, and consider BCDR aspects for dependent applications and other resources. These considerations will help ensure availability in the secondary disaster recovery location.</p>

Design area	Description
C	Depending on the size of your deployment and organization structure ensure all subscription have enough quota to run Azure Virtual Desktop workloads in different regions and that you have the right Azure RBAC roles assigned.
D	Through availability zones , VMs in the host pool are distributed across different datacenters. VMs are still in the same region, and they have higher resiliency and a higher formal 99.99 percent high-availability SLA . Your capacity planning should include sufficient extra compute capacity to ensure that Azure Virtual Desktop continues to operate, even if a single availability zone is lost.
E	Use FSLogix Cloud Cache to build profile resiliency for your users'. FSLogix Cloud Cache does impact the sign-on and sign out experience when using poor performing storage. It's common for environments using Cloud Cache to have slightly slower sign-on and sign out times, relative to using traditional VHDLocations, using the same storage. Review the FSLogix Cloud Cache documentation for recommendations regarding local cache storage.
F	Azure NetApp Files for enterprises offer the most value to customers. The Azure services simplify management for Azure Virtual Desktop and are the preferred storage solutions for this workload.
G	Storage options for FSLogix profile containers in Azure Virtual Desktop compares the different managed storage solutions that are available.
H	Separate user profile and Office container disks. FSLogix offers the option to place disks in separate storage locations.
I	For AppAttach disks and when needed, use Azure Storage built-in replication mechanisms for BCDR for environments that are less critical. Use zone-redundant storage (ZRS) or GRS for Azure Files.
J	To prevent user data from data loss or logical corruption use the Azure Backup to protect critical workloads.
K	The golden image that you use for host pool deployment in the secondary disaster recovery region should be the same you use for the primary. You should store images in the Azure Compute Gallery and configure multiple image replicas in both the primary and the secondary locations.

Failover and fallback

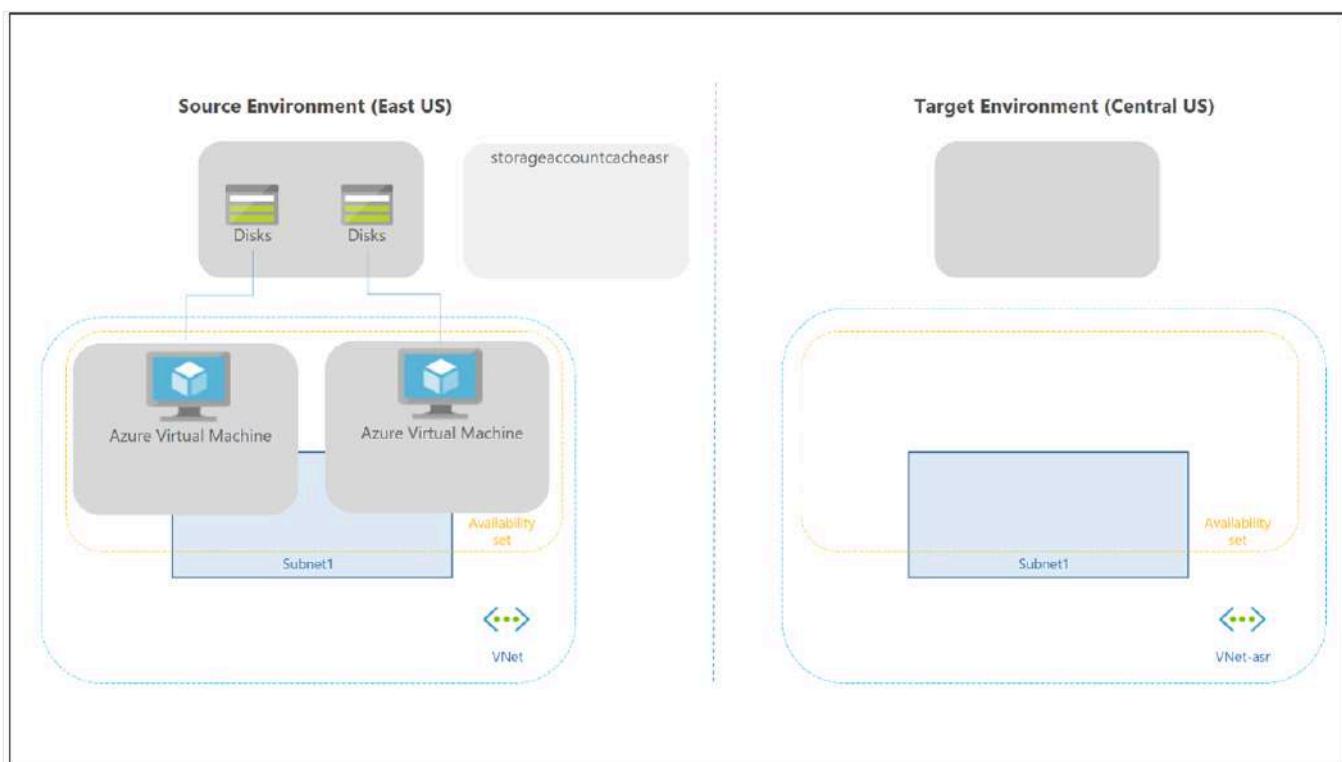
Personal host pool scenario

! Note

Only the active-passive model is covered in this section—an active-active doesn't require any failover or administrator intervention.

Failover and fallback for a personal host pool is different, as there's no Cloud Cache and external storage used for Profile and Office containers. You can still use FSLogix technology to save the data in a container from the session host. There's no secondary host pool in the disaster recovery region, so there's no need to create more workspaces and Virtual Desktop resources to replicate and align. You can use Site Recovery to replicate session host VMs.

You can use Site Recovery in several different scenarios. For Virtual Desktop, use the [Azure to Azure disaster recovery architecture in Azure Site Recovery](#).



The following considerations and recommendations apply:

- Site Recovery failover isn't automatic—an administrator must trigger it by using the Azure portal or [PowerShell/API](#).
- You can script and automate the entire Site Recovery configuration and operations by using [PowerShell](#).

- Site Recovery has a declared RTO inside its [service-level agreement \(SLA\)](#). Most of the time, Site Recovery can fail over VMs within minutes.
- You can use Site Recovery with Azure Backup. For more information, see [Support for using Site Recovery with Azure Backup](#).
- You must enable Site Recovery at the VM level, as there's no direct integration in the Virtual Desktop portal experience. You must also trigger failover and fallback at the single VM level.
- Site Recovery provides test failover capability in a separate subnet for general Azure VMs. Don't use this feature for Virtual Desktop VMs, since you would have two identical Virtual Desktop session hosts calling the service control plane at the same time.
- Site Recovery doesn't maintain Virtual Machine extensions during replication. If you enable any custom extensions for Virtual Desktop session host VMs, you must reenable the extensions after failover or fallback. The Virtual Desktop built-in extensions `joindomain` and `Microsoft.PowerShell.DSC` are only used when a session host VM is created. It's safe to lose them after a first failover.
- Be sure to review [Support matrix for Azure VM disaster recovery between Azure regions](#) and check requirements, limitations, and the compatibility matrix for the Site Recovery Azure-to-Azure disaster recovery scenario, especially the supported OS versions.
- When you fail over a VM from one region to another, the VM starts up in the target disaster recovery region in an unprotected state. Failback is possible, but the user must [reprotect](#) VMs in the secondary region, and then enable replication back to the primary region.
- Execute periodic testing of failover and fallback procedures. Then document an exact list of steps and recovery actions based on your specific Virtual Desktop environment.

Pooled host pool scenario

One of the desired characteristics of an active-active disaster recovery model is that administrator intervention isn't required to recover the service if there's an outage. Failover procedures should only be necessary in an active-passive architecture.

In an active-passive model, the secondary disaster recovery region should be idle, with minimal resources configured, and active. Configuration should be kept aligned with the primary region. If there's a failover, reassessments for all users to all desktop and application groups for remote apps in the secondary disaster recovery host pool happen at the same time.

It's possible to have an active-active model and partial failover. If the host pool is only used to provide desktop and application groups, then you can partition the users in multiple nonoverlapping Active Directory groups and reassign the group to desktop and application groups in the primary or secondary disaster recovery host pools. A user shouldn't have access to both host pools at the same time. If there's multiple application groups and applications, the

user groups you use to assign users might overlap. In this case, it's difficult to implement an active-active strategy. Whenever a user starts a remote app in the primary host pool, the user profile is loaded by FSLogix on a session host VM. Trying to do the same on the secondary host pool might cause a conflict on the underlying profile disk.

Warning

By default, FSLogix [registry settings](#) prohibit concurrent access to the same user profile from multiple sessions. In this BCDR scenario, you shouldn't change this behavior and leave a value of **0** for registry key **ProfileType**.

Here's the initial situation and configuration assumptions:

- The host pools in the primary region and secondary disaster recovery regions are aligned during configuration, including Cloud Cache.
- In the host pools, both DAG1 desktop and APPG2 and APPG3 remote app application groups are offered to users.
- In the host pool in the primary region, Active Directory user groups GRP1, GRP2, and GRP3 are used to assign users to DAG1, APPG2, and APPG3. These groups might have overlapping user memberships, but since the model here uses active-passive with full failover, it's not a problem.

The following steps describe when a failover happens, after either a planned or unplanned disaster recovery.

1. In the primary host pool, remove user assignments by the groups GRP1, GRP2, and GRP3 for application groups DAG1, APPG2, and APPG3.
2. There's a forced disconnection for all connected users from the primary host pool.
3. In the secondary host pool, where the same application groups are configured, you must grant user access to DAG1, APPG2, and APPG3 using groups GRP1, GRP2, and GRP3.
4. Review and adjust the capacity of the host pool in the secondary region. Here, you might want to rely on an autoscale plan to automatically power on session hosts. You can also manually start the necessary resources.

The **Failback** steps and flow are similar, and you can execute the entire process multiple times. Cloud Cache and configuring the storage accounts ensures that Profile and Office container data is replicated. Before failback, ensure that the host pool configuration and compute resources are recovered. For the storage part, if there's data loss in the primary region, Cloud Cache replicates Profile and Office container data from the secondary region storage.

It's also possible to implement a test failover plan with a few configuration changes, without affecting the production environment.

- Create a few new user accounts in Active Directory for production.
- Create a new Active Directory group named **GRP-TEST** and assign users.
- Assign access to DAG1, APPG2, and APPG3 by using the GRP-TEST group.
- Give instructions to users in the GRP-TEST group to test applications.
- Test the failover procedure by using the GRP-TEST group to remove access from the primary host pool and grant access to the secondary disaster recovery pool.

Important recommendations:

- Automate the failover process by using PowerShell, the Azure CLI, or another available API or tool.
- Periodically test the entire failover and failback procedure.
- Conduct a regular configuration alignment check to ensure host pools in the primary and secondary disaster region are in sync.

Backup

An assumption in this guide is that there's profile split and data separation between Profile containers and Office containers. FSLogix permits this configuration and the usage of separate storage accounts. Once in separate storage accounts, you can use different backup policies.

- For ODFC Container, if the content represents only cached data that can be rebuilt from on-line data store like Microsoft 365, it isn't necessary to back up data.
- If it's necessary to back up Office container data, you can use a less expensive storage or a different backup frequency and retention period.
- For a personal host pool type, you should execute the backup at the session host VM level. This method only applies if the data is stored locally.
- If you use OneDrive and known folder redirection, the requirement to save data inside the container might disappear.

Note

OneDrive backup is not considered in this article and scenario.

- Unless there's another requirement, backup for the storage in the primary region should be enough. Backup of the disaster recovery environment isn't normally used.
- For Azure Files share, use [Azure Backup](#).
 - For the vault [resiliency type](#), use zone-redundant storage if off-site or region backup storage isn't required. If those backups are required, use geo-redundant storage.

- Azure NetApp Files provides its own built-in [backup solution](#).
 - Make sure you check the region [feature availability](#), along with requirements and limitations.
- The separate storage accounts used for MSIX should also be covered by a backup if the application packages repositories can't be easily rebuilt.

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal authors:

- [Ben Martin Baur](#) | Senior Windows Cloud Technical Specialist
- [Igor Pagliai](#) | FastTrack for Azure (FTA) Principal Engineer

Other contributors:

- [Nelson Del Villar](#) | Cloud Solution Architect, Azure Core Infrastructure
- [Jason Martinez](#) | Technical Writer

Next steps

- [Virtual Desktop disaster recovery plan](#)
- [BCDR for Virtual Desktop - Cloud Adoption Framework](#)
- [Cloud Cache to create resiliency and availability](#)

Related resources

- [Design reliable Azure applications](#)
- [Azure files accessed on-premises and secured by AD DS](#)
- [Virtual Desktop for the enterprise](#)
- [Enterprise file shares with disaster recovery](#)
- [FSLogix configuration examples](#)

Azure Virtual Desktop for the enterprise

Microsoft Entra ID

Microsoft Entra

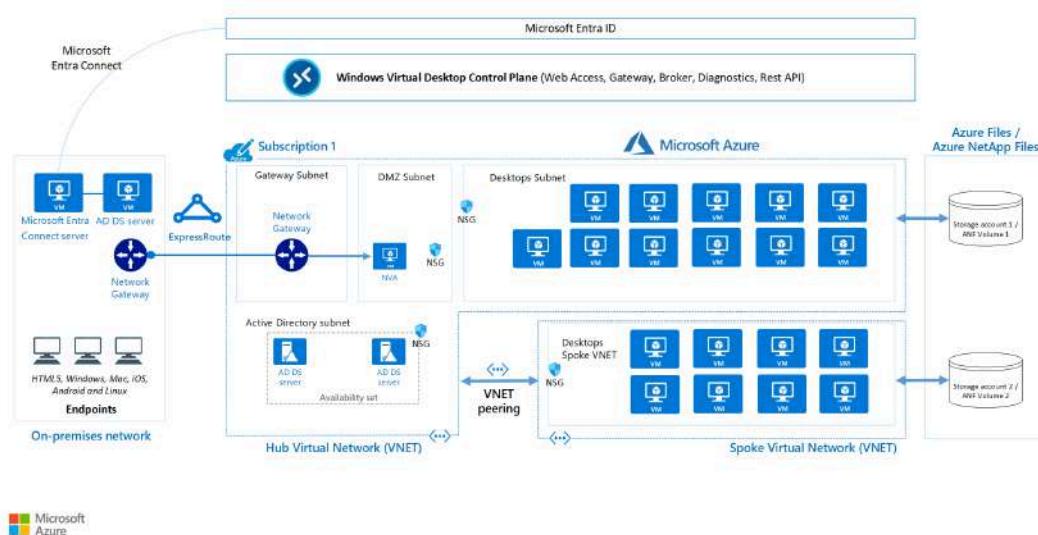
Azure Virtual Network

Azure Virtual Desktop

Azure Virtual Desktop [↗](#) is a desktop and application virtualization service that runs in Azure. This article is intended to help desktop infrastructure architects, cloud architects, desktop administrators, and system administrators explore Azure Virtual Desktop and build virtualized desktop infrastructure (virtual desktop infrastructure (VDI)) solutions at enterprise scale. Enterprise-scale solutions generally cover 1,000 or more virtual desktops.

Architecture

A typical architectural setup for Azure Virtual Desktop is illustrated in the following diagram:



Download a [Visio file ↗](#) of this architecture.

Dataflow

The diagram's dataflow elements are described here:

- The application endpoints are in a customer's on-premises network. Azure ExpressRoute extends the on-premises network into Azure, and Microsoft Entra Connect integrates the customer's Active Directory Domain Services (AD DS) with Microsoft Entra ID.
- The Azure Virtual Desktop control plane handles web access, gateway, broker, diagnostics, and extensibility components such as REST APIs.

- The customer manages AD DS and Microsoft Entra ID, Azure subscriptions, virtual networks, [Azure Files or Azure NetApp Files](#), and the Azure Virtual Desktop host pools and workspaces.
- To increase capacity, the customer uses two Azure subscriptions in a hub-spoke architecture and connects them via virtual network peering.

For more information about FSLogix Profile Container - Azure Files and Azure NetApp Files best practices, see [FSLogix configuration examples](#).

Components

Azure Virtual Desktop service architecture is similar to [Windows Server Remote Desktop Services \(RDS\)](#). Although Microsoft manages the infrastructure and brokering components, enterprise customers manage their own desktop host virtual machines (VMs), data, and clients.

Components that Microsoft manages

Microsoft manages the following Azure Virtual Desktop services, as part of Azure:

- **Web Access:** By using the [Web Access](#) service within Azure Virtual Desktop you can access virtual desktops and remote apps through an HTML5-compatible web browser just as you would with a local PC, from anywhere and on any device. You can secure web access by using multifactor authentication in Microsoft Entra ID.
- **Gateway:** The Remote Connection Gateway service connects remote users to Azure Virtual Desktop apps and desktops from any internet-connected device that can run an Azure Virtual Desktop client. The client connects to a gateway, which then orchestrates a connection from a VM back to the same gateway.
- **Connection Broker:** The Connection Broker service manages user connections to virtual desktops and remote apps. Connection Broker provides load balancing and reconnection to existing sessions.
- **Diagnostics:** Remote Desktop Diagnostics is an event-based aggregator that marks each user or administrator action on the Azure Virtual Desktop deployment as a success or failure. Administrators can query the event aggregation to identify failing components.
- **Extensibility components:** Azure Virtual Desktop includes several extensibility components. You can manage Azure Virtual Desktop by using Windows PowerShell or with the provided REST APIs, which also enable support from third-party tools.

Components that you manage

You manage the following components of Azure Virtual Desktop solutions:

- **Azure Virtual Network:** With [Azure Virtual Network](#), Azure resources such as VMs can communicate privately with each other and with the internet. By connecting Azure Virtual Desktop host pools to an Active Directory domain, you can define network topology to access virtual desktops and virtual apps from the intranet or internet, based on organizational policy. You can connect an Azure Virtual Desktop instance to an on-premises network by using a virtual private network (VPN), or you can use [Azure ExpressRoute](#) to extend the on-premises network into Azure over a private connection.
- **Microsoft Entra ID:** Azure Virtual Desktop uses [Microsoft Entra ID](#) for identity and access management. Microsoft Entra integration applies Microsoft Entra security features, such as Conditional Access, multifactor authentication, and [Intelligent Security Graph](#), and it helps maintain app compatibility in domain-joined VMs.
- **Active Directory Domain Services (Optional):** Azure Virtual Desktop VMs can either be domain joined to an [AD DS](#) service or use [Deploy Microsoft Entra joined virtual machines in Azure Virtual Desktop](#)
 - When using an AD DS domain, the domain must be in sync with Microsoft Entra ID to associate users between the two services. You can use [Microsoft Entra Connect](#) to associate AD DS with Microsoft Entra ID.
 - When using Microsoft Entra join, review the [supported configurations](#) to ensure your scenario is supported.
- **Azure Virtual Desktop session hosts:** Session hosts are VMs that users connect to for their desktops and applications. Several versions of Windows are supported and you can create images with your applications and customizations. You can choose VM sizes, including GPU-enabled VMs. Each session host has an Azure Virtual Desktop host agent, which registers the VM as part of the Azure Virtual Desktop workspace or tenant. Each host pool can have one or more app groups, which are collections of remote applications or desktop sessions that you can access. To see which versions of Windows are supported, see [Operating systems and licenses](#).
- **Azure Virtual Desktop workspace:** The Azure Virtual Desktop workspace or tenant is a management construct for managing and publishing host pool resources.

Scenario details

Potential use cases

The greatest demand for enterprise virtual desktop solutions comes from:

- Security and regulation applications, such as financial services, healthcare, and government.
- Elastic workforce needs, such as remote work, mergers and acquisitions, short-term employees, contractors, and partner access.
- Specific employees, such as bring your own device (BYOD) and mobile users, call centers, and branch workers.
- Specialized workloads, such as design and engineering, legacy apps, and software development testing.

Personal and pooled desktops

By using personal desktop solutions, sometimes called *persistent desktops*, users can always connect to the same specific session host. Users can ordinarily modify their desktop experience to meet personal preferences, and they can save files in the desktop environment. Personal desktop solutions:

- Let users customize their desktop environment, including user-installed applications, and users can save files within the desktop environment.
- Allow assigning dedicated resources to specific users, which can be helpful for some manufacturing or development use cases.

Pooled desktop solutions, also called *non-persistent desktops*, assign users to whichever session host is currently available, depending on the load-balancing algorithm. Because users don't always return to the same session host each time they connect, they have limited ability to customize the desktop environment and don't usually have administrator access.

ⓘ Note

Persistent and non-persistent terminology in this case is in reference to the persistence of the user profile. It does not imply that the operating system disk reverts to a golden image or discards changes on reboot.

Windows servicing

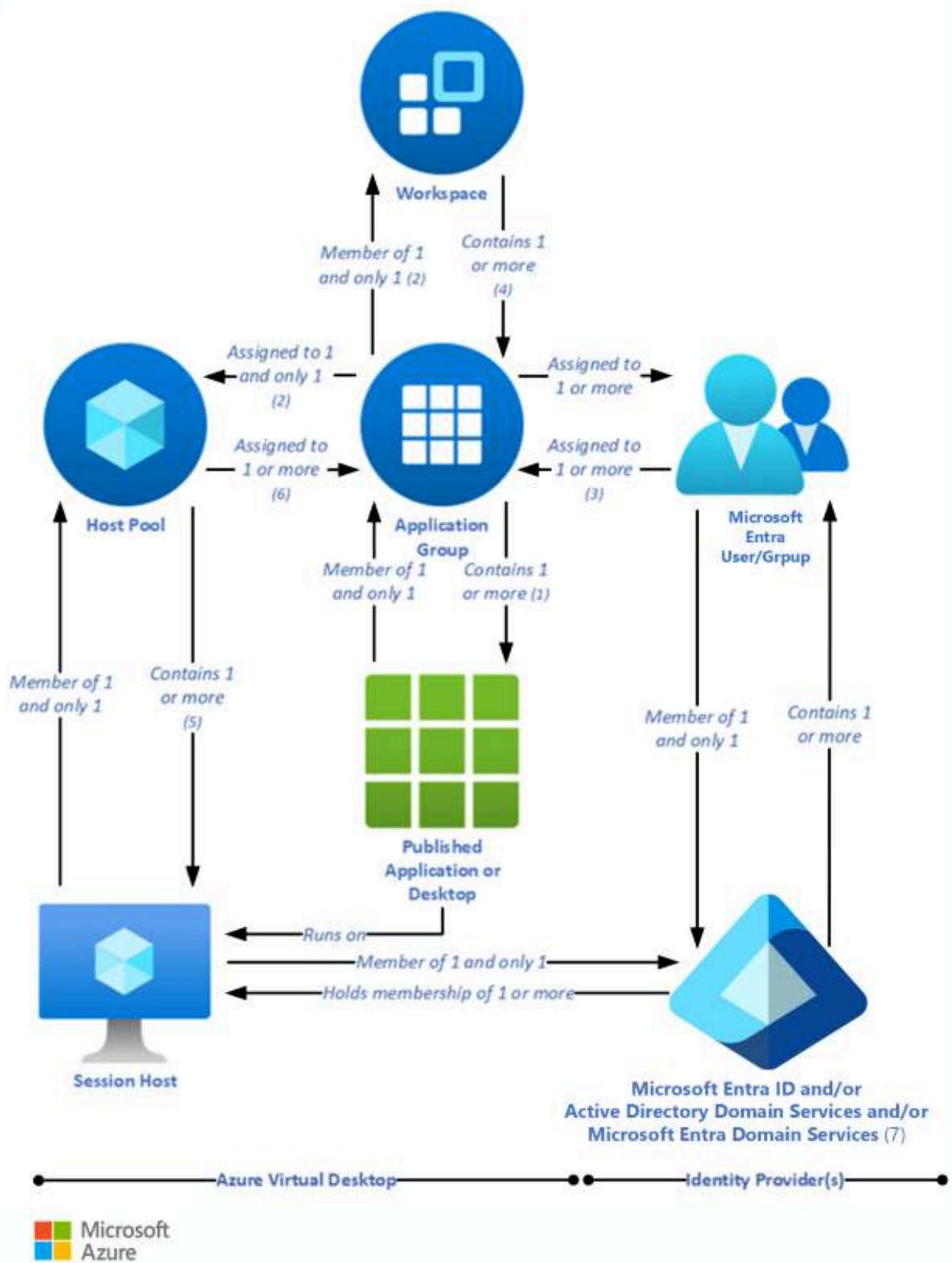
There are several options for updating Azure Virtual Desktop instances. Deploying an updated image every month guarantees compliance and state.

- [Microsoft Endpoint Configuration Manager \(MECM\)](#) updates server and desktop operating systems.

- [Windows Updates for Business](#) updates desktop operating systems such as Windows 11 Enterprise multi-session.
- [Azure Update Management](#) updates server operating systems.
- [Azure Log Analytics](#) checks compliance.
- Deploy a new (custom) image to session hosts every month for the latest Windows and applications updates. You can use an image from Azure Marketplace or a [custom Azure-managed image](#).

Relationships between key logical components

The relationships between host pools, workspaces, and other key logical components vary. They're summarized in the following diagram:



The numbers in the following descriptions correspond to those in the preceding diagram.

- (1) An application group that contains a published desktop can only contain MSIX packages mounted to the host pool (the packages will be available in the *Start* menu of

the session host), it can't contain any other published resources and is called a desktop application group.

- (2) Application groups assigned to the same host pool must be members of the same workspace.
- (3) A user account can be assigned to an application group either directly or via a Microsoft Entra group. It's possible to assign no users to an application group, but then it can't service any.
- (4) It's possible to have an empty workspace, but it can't service users.
- (5) It's possible to have an empty host pool, but it can't service users.
- (6) It's possible for a host pool not to have any application groups assigned to it but it can't service users.
- (7) Microsoft Entra ID is required for Azure Virtual Desktop. This is because Microsoft Entra user accounts and groups must always be used to assign users to Azure Virtual Desktop application groups. Microsoft Entra ID is also used to authenticate users into the Azure Virtual Desktop service. Azure Virtual Desktop session hosts can also be members of a Microsoft Entra domain, and in this situation the Azure Virtual Desktop-published applications and desktop sessions will also be launched and run (not just assigned) by using Microsoft Entra accounts.
 - (7) Alternatively, Azure Virtual Desktop session hosts can be members of an AD DS domain, and in this situation the Azure Virtual Desktop-published applications and desktop sessions will be launched and run (but not assigned) by using AD DS accounts. To reduce user and administrative overhead, AD DS can be synchronized with Microsoft Entra ID through Microsoft Entra Connect.
 - (7) Finally, Azure Virtual Desktop session hosts can, instead, be members of a Microsoft Entra Domain Services domain, and in this situation the Azure Virtual Desktop-published applications and desktop sessions will be launched and run (but not assigned) by using Microsoft Entra Domain Services accounts. Microsoft Entra ID is automatically synchronized with Microsoft Entra Domain Services, one way, from Microsoft Entra ID to Microsoft Entra Domain Services only.

 [Expand table](#)

Resource	Purpose	Logical relationships
Published desktop	A Windows desktop environment that runs on Azure Virtual Desktop session hosts and is delivered to users over the network	Member of one and only one application group (1)

Resource	Purpose	Logical relationships
Published application	A Windows application that runs on Azure Virtual Desktop session hosts and is delivered to users over the network	Member of one and only one application group
Application group	A logical grouping of published applications or a published desktop	<ul style="list-style-type: none"> - Contains a published desktop (1) or one or more published applications - Assigned to one and only one host pool (2) - Member of one and only one workspace (2) - One or more Microsoft Entra user accounts or groups are assigned to it (3)
Microsoft Entra user account/group	Identifies the users who are permitted to launch published desktops or applications	<ul style="list-style-type: none"> - Member of one and only one Microsoft Entra ID - Assigned to one or more application groups (3)
Microsoft Entra ID (7)	Identity provider	<ul style="list-style-type: none"> - Contains one or more user accounts or groups, which must be used to assign users to application groups, and can also be used to sign in to the session hosts - Can hold the memberships of the session hosts - Can be synchronized with AD DS or Microsoft Entra Domain Services
AD DS (7)	Identity and directory services provider	<ul style="list-style-type: none"> - Contains one or more user accounts or groups, which can be used to sign in to the session hosts - Can hold the memberships of the session hosts - Can be synchronized with Microsoft Entra ID
Microsoft Entra Domain Services (7)	Platform as a service (PaaS)-based identity and directory services provider	<ul style="list-style-type: none"> - Contains one or more user accounts or groups, which can be used to sign in to the session hosts - Can hold the memberships of the

Resource	Purpose	Logical relationships
		session hosts - Synchronized with Microsoft Entra ID
Workspace	A logical grouping of application groups	Contains one or more application groups (4)
Host pool	A group of identical session hosts that serve a common purpose	- Contains one or more session hosts (5) - One or more application groups are assigned to it (6)
Session host	A virtual machine that hosts published desktops or applications	Member of one and only one host pool

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

Use the [assessment tool](#) to assess the readiness of your Azure Virtual Desktop workload. This tool checks your alignment to best practices described in the [Azure Virtual Desktop workload documentation](#).

Reliability

Reliability helps ensure that your application can meet the commitments that you make to your customers. For more information, see [Design review checklist for Reliability](#).

- **Ensure capacity is reserved:** To ensure guaranteed allocation of compute resources, you can request an [On-demand capacity reservation](#) with no term commitment and can be combined with reserved instances.
- **Add Intra-region resiliency:** Use [Availability zones](#) for Azure services that support them such as:
 - Virtual Machines (Session Hosts)
 - Azure Storage (FSLogix or App Attach). For more information, see [Azure Storage redundancy](#).

- **Build a business continuity plan:** If [availability zones](#) do not meet your RTO or RPO targets, review the guidance on [Multiregion Business Continuity and Disaster Recovery \(BCDR\)](#) for Azure Virtual Desktop.

Security

Security provides assurances against deliberate attacks and the misuse of your valuable data and systems. For more information, see [Design review checklist for Security](#).

Consider the following security-related factors when you deploy Azure Virtual Desktop.

- **Use Microsoft Entra ID:** Users can sign into Azure Virtual Desktop from anywhere using different devices and clients. To minimize the risk of unauthorized access and provide your organization with the ability to manage sign-in risks, [Enforce Microsoft Entra multifactor authentication using Conditional Access](#).
- **Use encryption:** By default, most Azure managed disks are encrypted at rest when persisting to the cloud. If your session hosts require more extensive encryption, like end-to-end encryption, review the guidance on [managed disk encryption options](#) to protect stored data from unauthorized access.
- **Use private networking:** If you require private connectivity to Azure Virtual Desktop resources, use [Azure Private Link with Azure Virtual Desktop](#) to constrain traffic between your virtual network and the service on the Microsoft Network.

 **Note**

For more security recommendations, see the guidance on [Security recommendations for Azure Virtual Desktop](#).

Cost Optimization

Cost Optimization focuses on ways to reduce unnecessary expenses and improve operational efficiencies. For more information, see [Design review checklist for Cost Optimization](#).

Consider the following cost-related factors when you deploy Azure Virtual Desktop.

- **Plan multi-session support:** For workloads with identical compute requirements, generally pooled host pools, [Windows Enterprise multi-session](#) offers the ability to accept more users to sign in to a single VM at once; reducing costs and administrative overhead.
- **Optimize licensing:** If you have Software Assurance, you can use [Azure Hybrid Benefit](#) to reduce the cost of your Azure compute infrastructure.

- **Pre-purchase compute:** You can commit to one-year or three-year plans, [Azure Reservations](#), based on your VM usage to receive a discount to significantly reduce your resource cost. This can be combined with Azure Hybrid Benefit for additional savings.
- **Scale in and out as needed:** If committing to Azure Reservations is not appropriate for your current needs, consider [Autoscale scaling plans](#) for dynamic provisioning/deprovisioning of session hosts as the demand changes through the day/week.
- **Evaluate load-balancing options:** Configure your host pool load balancing algorithm to depth-first. Be aware however, this can configuration degrades the users experience; the default breadth-first optimized user experience. For more information, see [Configure host pool load balancing in Azure Virtual Desktop](#).

Operational Excellence

Operational Excellence covers the operations processes that deploy an application and keep it running in production. For more information, see [Design review checklist for Operational Excellence](#).

- **Configure alerts:** Configure [Service Health](#) and [Resource Health](#) alerts to stay informed about the health of the Azure services and regions that you use.
 - Monitor the Azure Storage solution that you use for hosting FSLogix Profiles or App Attach shares to ensure that thresholds aren't exceeded, which might have a negative impact on your user experience.
- **Collect performance data:** Install the [Azure Monitoring Agent](#) on your Azure Virtual Desktop session hosts to extract and monitor performance counters and event logs. For more information, see the [list of configurable performance metrics/counters and event logs](#).
- **Collect usage insights:** Use [Azure Virtual Desktop Insights](#) to help with checks such as which client versions are connecting, opportunities for cost saving, or knowing if you have resource limitations or connectivity issues.
- **Tune diagnostic settings:** Enable diagnostic settings for all services, Azure Virtual Desktop workspaces, application groups, host pools, storage accounts. Determine which settings are meaningful to your operations. Turn off settings that aren't meaningful to avoid undue costs; storage accounts (specifically the file service) that see a high amount of IOPS can incur high monitoring costs.

Performance Efficiency

Performance Efficiency refers to your workload's ability to scale to meet user demands efficiently. For more information, see [Design review checklist for Performance Efficiency](#).

- **Use antivirus exclusions:** For profile solutions like FSLogix that mount virtual hard disk files, it's recommended to exclude those file extensions. For more information, see [Configure Antivirus file and folder exclusions](#).
- **Tune for latency:** For clients using a Point-to-Site (P2S) VPN connection use a split tunnel that's based on User Datagram Protocol (UDP) to reduce latency and optimize your tunnel bandwidth usage. For on-site clients who use a VPN or Azure ExpressRoute, use [RDP Shortpath](#) to reduce the round-trip time, which improves the user experience in latency-sensitive applications and input methods.
- **Use right-size compute:** [Virtual machine sizing guidelines](#) lists the maximum suggested number of users per virtual central processing unit (vCPU) and minimum VM configurations for different workloads. This data helps estimate the VMs you need in your host pool.
 - Utilize simulation tools to test deployments with both stress tests and real-life usage simulations. Make sure that the system is responsive and resilient enough to meet user needs and remember to vary the load sizes when testing.
- **Use ephemeral OS disks:** If you treat your session hosts like cattle as opposed to pets, [Ephemeral OS disks](#) are great way to improve performance, latency similar to temporary disks, and simultaneously save costs as they are free.

Limitations

Azure Virtual Desktop, much like Azure, has certain service limitations that you need to be aware of. To avoid having to make changes in the scaling phase, it's a good idea to address some of these limitations during the design phase.

For more information about the Azure Virtual Desktop Service limitations, see [Azure Virtual Desktop Service limits](#).

Also, note that:

- You can't create more than 500 application groups per single Microsoft Entra tenant*.
 - If you require more than 500 application groups, submit a support ticket via the Azure portal.
- We recommend that you do *not* publish more than 50 applications per application group.
- We recommend that you deploy no more than 5,000 VMs per Azure subscription per region. This recommendation applies to both personal and pooled host pools, based on Windows Enterprise single and multi-session. Most customers use Windows Enterprise multi-session, which allows multiple users to sign in to each VM. You can increase the resources of individual session-host VMs to accommodate more user sessions.
- For automated session-host scaling tools, the limits are around 2,500 VMs per Azure subscription per region, because VM status interaction consumes more resources.

- To manage enterprise environments with more than 5,000 VMs per Azure subscription in the same region, you can create multiple Azure subscriptions in a hub-spoke architecture and connect them via virtual network peering (using one subscription per spoke). You could also deploy VMs in a different region in the same subscription to increase the number of VMs.
- Azure Resource Manager subscription API throttling limits don't allow more than 600 Azure VM reboots per hour via the Azure portal. You can reboot all your machines at once via the operating system, which doesn't consume any Azure Resource Manager subscription API calls. For more information about counting and troubleshooting throttling limits based on your Azure subscription, see [Troubleshoot API throttling errors](#).
- You can currently deploy up to 132 VMs in a single ARM template deployment in the Azure Virtual Desktop portal. To create more than 132 VMs, run the ARM template deployment in the Azure Virtual Desktop portal multiple times.
- Azure VM session-host name prefixes can't exceed 11 characters, due to auto-assigning of instance names and the NetBIOS limit of 15 characters per computer account.
- By default, you can deploy up to 800 instances of most resource types in a resource group. Azure Compute doesn't have this limit.

For more information about Azure subscription limitations, see [Azure subscription and service limits, quotas, and constraints](#).

Deploy this scenario

A collection of [ARM templates](#) can be employed to automate the deployment of your Azure Virtual Desktop environment. These ARM templates support only the Azure Resource Manager Azure Virtual Desktop objects. These ARM templates don't support Azure Virtual Desktop (classic).

More scenarios are available from Microsoft Developer Tools which supports several deployment options:

- [Azure Virtual Desktop with Microsoft Entra ID Join](#)
- [Azure Virtual Desktop with FSLogix and AD DS Join](#)

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal author:

- [Tom Hickling](#) | Senior Product Manager, Azure Virtual Desktop Engineering

Other contributor:

- [Nelson Del Villar](#) | Cloud Solution Architect, Azure Core Infrastructure

Next steps

- [Azure Virtual Desktop partner integrations](#) lists approved Azure Virtual Desktop partner providers and independent software vendors.
- Use the [Virtual Desktop Optimization Tool](#) to help optimize performance in a Windows 11 Enterprise VDI (virtual desktop infrastructure) environment.
- For more information, see [Deploy Microsoft Entra joined virtual machines in Azure Virtual Desktop](#).
- Learn more about [Active Directory Domain Services](#).
- [What is Microsoft Entra Connect?](#)
- Learn more about the [Azure Virtual Desktop Well-Architected Framework](#)

Deploy Esri ArcGIS Pro in Azure Virtual Desktop

Azure Virtual Desktop

Azure NetApp Files

Azure Monitor

Azure Policy

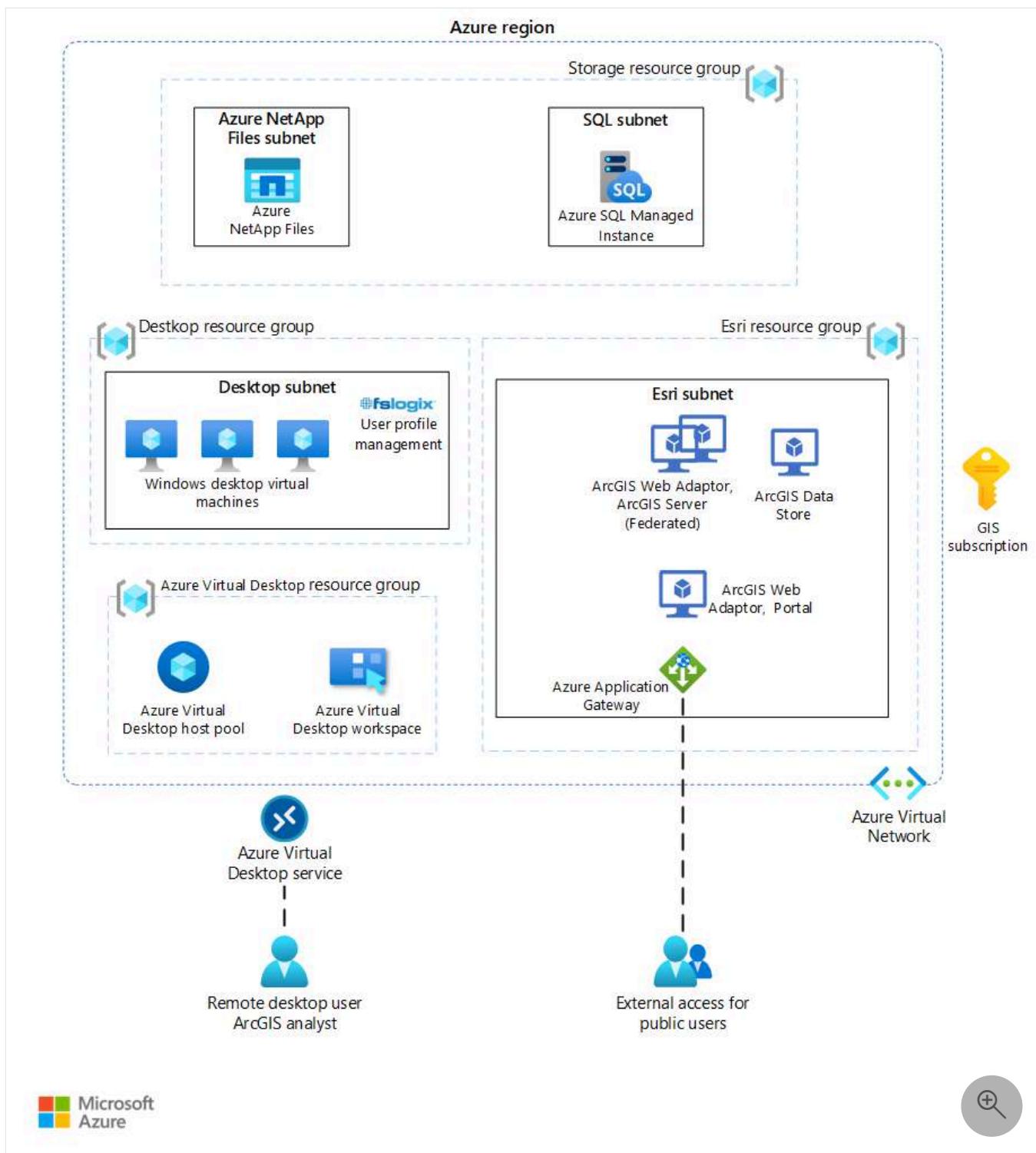
Microsoft Entra ID

This architecture shows how you can deploy Esri ArcGIS Pro in Azure Virtual Desktop to support the hyperscale of Azure. The architecture also includes back-end components like ArcGIS Enterprise to build a complete system on Azure.

ArcGIS® is a trademark of its company. No endorsement is implied by the use of this mark.

Architecture

The following diagram presents a high-level architecture for deploying ArcGIS components on Azure.



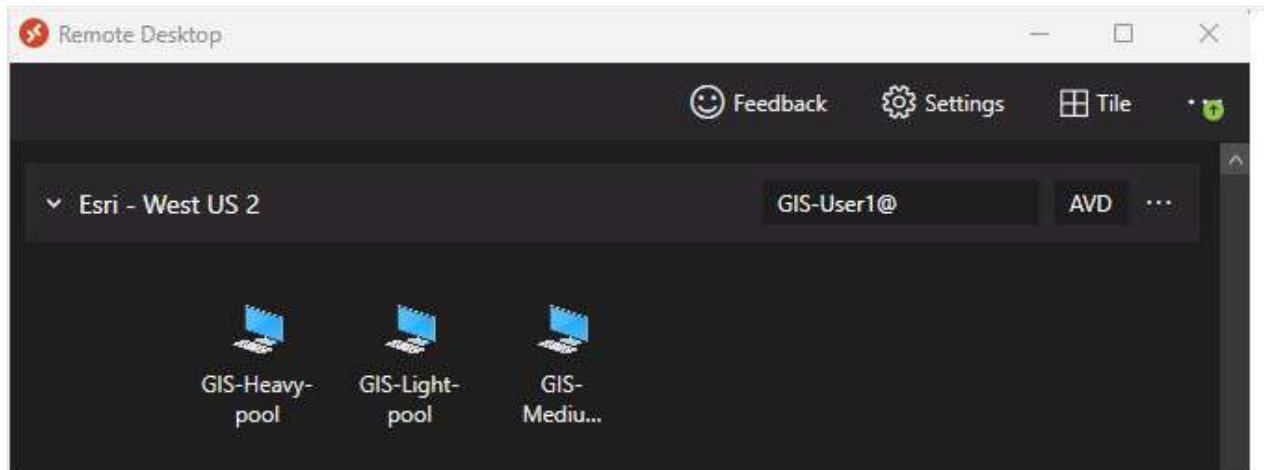
Download a [Visio file](#) of this architecture.

Workflow

- This solution is deployed to a single region with storage, GIS desktop, GIS back end, and Azure Virtual Desktop resource groups. Each resource group contains one subnet, and all subnets are in one virtual network. All components are in a single Azure subscription. This architecture is a three-tier deployment.
- The application endpoints are in the on-premises network.

- The Azure Virtual Desktop control plane handles web access, gateway, broker, diagnostics, and extensibility components like REST APIs.
- You manage Microsoft Entra Domain Services and Microsoft Entra ID, Azure subscriptions, virtual networks, [Azure Files or Azure NetApp Files](#), and the Azure Virtual Desktop host pools and workspaces.
- GIS analysts, administrators, and editors connect to Azure Virtual Desktop via a Remote Desktop Protocol (RDP) session. From there, ArcGIS Pro is accessed and takes advantage of the GPUs for light, medium, and heavy workflows. *Light* refers to a 2D workflow, *medium* refers to a more demanding 2D workflow, and *heavy* refers to a 2D or 3D workflow that requires GPUs. GIS administrators can also use ArcGIS Pro to publish services and administer the enterprise geodatabase. Finally, GIS editors can maintain the vector and raster layers.

Administrators can also make it possible to publish new versions of ArcGIS Pro by using semantic versioning. For example, as new versions of ArcGIS Pro are available, like ArcGIS Pro 3.0, the new version can be published in the Remote Desktop tool. As a result, users can pick that new version when they're ready to upgrade without having to perform the upgrade themselves. The GPU drivers can be included in the creation of the images that the deployments are based on.



- Web GIS users can also take advantage of this solution by accessing ArcGIS Enterprise administrative interfaces either in the browser in the Azure Virtual Desktop RDP session or via their local browser (if ArcGIS is published as public facing). The Azure application gateway routes the traffic to the correct endpoint for the ArcGIS server roles. As with ArcGIS Pro, the latency between the browsers and the back end are minimized.
- You can deploy the enterprise geodatabase in Azure SQL Managed Instance. ArcGIS Pro users can then create, manage, and edit the geodatabase from an RDP session. During the creation of the Azure Virtual Desktop image, administrators can include the ODBC drivers so users don't have to install them on the Azure Virtual Desktop VMs.

- Azure NetApp Files supports fast access to the ArcGIS Server configuration store and directories. You can use Azure Files and Azure Storage, but Azure NetApp Files costs less for large deployments. Additionally, you can use Azure NetApp Files to store Portal for ArcGIS items and raster images, lidar data, and so on.

Components

- [Azure NetApp Files](#) is an enterprise-class, high-performance, metered file Network-attached storage (NAS) service. In this architecture, it stores ArcGIS Server configuration data, raster images, lidar datasets, and other geospatial files.
- [Azure Monitor](#) is a collection of tools that provides visibility into the state of your system. In this architecture, Azure Monitor provides visibility into system performance and helps identify and resolve problems that affect your workload's components.
- [Azure Policy](#) is a governance tool that enforces rules and standards across Azure resources. In this architecture, it ensures compliance with workload policies, such as resource tagging, location restrictions, and security configurations. Through its compliance dashboard, it provides an aggregated view of the overall state of the environment and the ability to drill down to per-resource, per-policy granularity.
- [Microsoft Entra ID](#) is an enterprise identity service that provides single sign-on, multifactor authentication, and other identity services to protect against cybersecurity threats. In this architecture, it secures user access to Azure Virtual Desktop and other services.
- [Active Directory Domain Services \(AD DS\)](#) is a directory service that provides traditional domain-based identity services like group policies and Kerberos authentication. It stores directory data and makes that data available to network users and administrators. AD DS stores information about user accounts, like names, passwords, and phone numbers, and enables other authorized users on the same network to access that information. This data store, also known as the *directory*, contains information about Windows Server Active Directory (Windows Server AD) objects. These objects typically include shared resources like servers, volumes, printers, and the network user and computer accounts. In this architecture, AD DS serves as the centralized identity and access control system. It authenticates users and devices, enforces security policies, and enables secure, single sign-on access to network resources.

Security is integrated with Windows Server AD through sign-in authentication and controlled access to objects in the directory. With a single network sign-in, administrators can manage directory data and organization throughout their network, and authorized network users can access resources anywhere on the network.

- [Azure Virtual Desktop](#) is a desktop and application virtualization service that delivers Windows desktops and apps remotely. In this architecture, Azure Virtual Desktop hosts ArcGIS Pro on GPU-enabled VMs so that users can run intensive GIS workflows from anywhere.
- [Azure SQL Managed Instance](#) is a managed SQL Server instance that includes built-in high availability and scalability. In this architecture, SQL Managed Instance stores the enterprise geodatabase, which enables ArcGIS Pro users to manage and edit spatial data in a secure and scalable environment.
- [Azure Application Gateway](#) is an application delivery controller-as-a-service offering that provides layer-7 load balancing, security, and web application firewall functionality. In this architecture, it distributes incoming requests to ArcGIS Server roles, which ensures efficient traffic routing and protection against common web vulnerabilities.
- [FSLogix](#) is a profile container solution that improves the user experience in virtual desktop environments. In this architecture, it enables fast logins and persistent user profiles for Azure Virtual Desktop users. It also allows users to roam between remote computing session hosts and optimize file input/output (I/O) between the host or client and the remote profile store.

For more information about FSLogix Profile Container, Azure Files, and Azure NetApp Files best practices, see [FSLogix configuration examples](#).

- [Azure Virtual Network](#) is a private cloud-based network that you can use to build your own secure network infrastructure in Azure. It enables secure communication between Azure resources through private IP addresses. In this architecture, Virtual Network connects all components, such as virtual desktops, databases, and storage, within a secure and isolated network.
- [ArcGIS Pro](#) is Esri's professional desktop GIS application for spatial analysis, mapping, and data editing. In this architecture, it runs on GPU-enabled Azure Virtual Desktop VMs, which allows users to perform advanced 2D and 3D geospatial tasks and publish services. It runs best on Azure high-performance computing VMs, like the NV-Series. You can scale the use of ArcGIS by using Azure Virtual Desktop.
- [ArcGIS Enterprise](#) is a comprehensive GIS platform for managing and sharing spatial data and services. In this architecture, you can add ArcGIS Enterprise to extend capabilities for hosting maps, apps, and spatial analytics across the organization. It works with ArcGIS Pro or ArcGIS Desktop (not included here because ArcGIS Pro replaces it).
- [Portal for ArcGIS](#) is a web-based interface for sharing and managing GIS content within ArcGIS Enterprise. In this architecture, it enables users to create, organize, and share

maps, scenes, and apps securely within the organization. Portal for ArcGIS is part of the base deployment.

- [ArcGIS Server](#) is back-end server software that's deployed with ArcGIS Enterprise or in a standalone deployment with ArcGIS Enterprise. In this architecture, it handles requests from users and applications, such as to draw maps, run tools, or query data. Its configuration and data is stored in Azure NetApp Files. It also has a management plane that enables administrators to start, stop, and delete services.
- [ArcGIS Server configuration store](#) is a shared file system that stores ArcGIS Server site configuration, including service definitions, logs, and settings. In this architecture, it resides on Azure NetApp Files to ensure high availability and performance for ArcGIS Server operations. It contains system configuration information so that, as ArcGIS Server scales to other machines, it can share that information.
- [Enterprise geodatabase](#) is a multi-user spatial database that supports versioning, replication, and advanced data models. You can deploy this database in many database management systems. In this architecture, it's hosted in SQL Managed Instance and serves as the authoritative data source for ArcGIS Pro and other GIS tools.

Alternatives

- You can use [ArcGIS Enterprise Builder](#) to set up a base ArcGIS Enterprise deployment on a single machine or multiple machines.
- Although Azure Files and Azure Blob Storage are fine for many enterprises, Azure NetApp Files might be better suited for GIS because of large raster image files, Portal for ArcGIS items, shapefiles, lidar datasets, file geodatabases, and other geospatial data types that require fast access.
- You can add other ArcGIS Enterprise server roles, like Raster Analytics Server, GeoAnalytics Server, GeoEvent Server, Knowledge Server, and Mission Server, to this base deployment as needed. You can also use newer technologies, like ArcGIS Enterprise on Kubernetes, as a replacement for or supplement to ArcGIS Enterprise. GPU-based VMs for Drone2Map, CityEngine, and SURE for ArcGIS can also take advantage of these VMs. For more information, see [ArcGIS Enterprise server roles](#).
- To increase capacity, you can use multiple Azure subscriptions in a hub-and-spoke architecture and connect them via virtual network peering. Also, you can use Azure landing zones to lay down the initial services. For more information, see [What is an Azure landing zone?](#).

Scenario details

Esri's technology is a geographic information system (GIS) that contains capabilities for the visualization, analysis, and data management of geospatial data. Esri's core technology is called *the ArcGIS platform*. It includes capabilities for mapping, spatial analysis, 3D GIS, imagery and remote sensing, data collection and management, and field operations. For more information, see the [ArcGIS page](#) on the Esri website.

A desktop app called *ArcGIS Pro* is a key part of the technology. It's a 64-bit professional desktop GIS. GIS analysts can use it to perform spatial analysis and edit spatial data. GIS administrators can use it to create and publish geospatial services.

Potential use cases

Esri's ArcGIS and virtual desktop solutions are frequently used for:

- Security and regulation applications like utilities (energy), healthcare, and government.
- Elastic workforce needs like remote work, mergers and acquisition, short-term employees, contractors, and partner access.
- Employees like bring your own device (BYOD) users, mobile users, and branch workers.
- Specialized workloads like land management (facilities and real estate), design and engineering, legacy apps, and software testing.

Although GIS has been implemented in Azure for many years, it has typically included only the back-end components. That implementation introduces latency between the client and server components. Organizations have been able to deploy desktop GIS on VMs from Azure Marketplace, but that deployment is for thick clients and isn't very scalable. This architecture addresses both challenges.

Considerations

These considerations implement the pillars of the Azure Well-Architected Framework, which is a set of guiding tenets that you can use to improve the quality of a workload. For more information, see [Well-Architected Framework](#).

Performance Efficiency

Performance Efficiency refers to your workload's ability to scale to meet user demands efficiently. For more information, see [Design review checklist for Performance Efficiency](#).

Ideally, the latency between the end user and the RDP session needs to be around 200 ms or less. This latency helps to ensure that, when ArcGIS Pro users interact with maps and perform measurements or edits, the interactive edits and the tooltips appear quickly enough. The [Azure Virtual Desktop Experience Estimator](#) can provide a quick assessment of connection round-

trip time (RTT) from your location, through the Azure Virtual Desktop service, and to each Azure region in which you can deploy virtual machines.

When you use a remote Windows session, your network's available bandwidth greatly affects the quality of your experience. The following table lists the minimum recommended bandwidths for a smooth user experience. These recommendations are based on the guidelines in [Remote Desktop workloads](#).

 [Expand table](#)

Workload type	Recommended bandwidth
Light	1.5 Mbps
Medium	3 Mbps
Heavy	5 Mbps
Power	15 Mbps

Keep in mind that the stress put on your network depends on both your app workload's output frame rate and your display resolution. If either the frame rate or display resolution increases, the bandwidth requirement also rises. For example, a light workload with a high-resolution display requires more available bandwidth than a light workload with regular or low resolution.

Ideally, all components in the preceding architecture diagram are deployed in a single region to minimize latency between components. However, for large organizations, a multi-region deployment is necessary and supported. Another component to consider is [Azure Front Door](#), which routes users to the closest region.

Another significant benefit of this architecture is that the latency between it and Esri's SaaS offerings, like ArcGIS Velocity and ArcGIS Image, is also reduced for ArcGIS Pro users and web browser users. All components of the ArcGIS platform are in the cloud.

Scalability

You can scale this architecture in many ways. You can scale the VMs for the back end or the desktops (both CPU and GPUs) in, out, up, or down. You can also deploy Azure Virtual Desktop on individual VMs or multi-session VMs. Azure Virtual Desktop can scale hundreds or

thousands of VMs. For more information, see [Windows 10 or Windows 11 Enterprise multisession remote desktops](#).

Testing

You can test your system's latency by using the [Connection Experience Indicator](#). You can use Esri's [ArcGIS Pro Performance Assessment Tool](#) to test the performance. Esri also recommends [tools for testing ArcGIS Enterprise](#). [Azure Load Testing](#) can also be helpful.

ArcGIS Pro virtual machine sizing guidelines for Azure Virtual Desktop and Remote Desktop Services

Whether you're running your session host virtual machines on Remote Desktop Services or Azure Virtual Desktop, different types of workloads require different virtual machine configurations. The examples in this article are generic guidelines, and you should only use them for initial performance estimates. For the best possible experience, optimize and scale your deployment depending on your users' needs.

ArcGIS Pro should use Windows 10 and Windows 11 multisession VMs to provide additional flexibility and greater return on investment. It is necessary to allocate the appropriate VM types to give each user enough resources such as GPU, CPU, and RAM. Always consider the number of connections and limit the simultaneous user access to each VM to avoid oversaturation and hindering performance.

Workloads

Users can run different types of workloads on the session host virtual machines. The following table shows examples of a range of workload types to help you estimate what size your virtual machines need to be. After you set up your virtual machines, continually monitor their actual usage and adjust their size accordingly. If you end up needing a bigger or smaller virtual machine, scale your existing deployment up or down.

The following table describes each ArcGIS workload. *Example users* are the types of users that might find each workload most helpful.

 [Expand table](#)

Workload type	Example user workflows	Activity
Light	Simple 2-D map display, navigation, and querying. Combining and presenting data prepared by others.	Viewing
Medium	2-D and 3-D map display, navigation, querying, and editing. Moderate use of GP tools. Compilation of presentation of data from multiple sources into a simple map layout.	Editing
Heavy	2-D and 3-D map display, navigation, querying, and editing. Advanced use of symbology including transparency, and dynamic labeling. Heavy 2-D and 3-D analysis involving visibility, and line of sight.	Visualizing

Review the [ArcGIS Pro 3.3 system requirements](#) and [ArcGIS Pro on Microsoft Azure Cloud](#) recommendations to complement your sizing effort.

Single-session recommendations

Single-session scenarios are when there's only one user signed in to a session host virtual machine at any one time. For example, if you use personal host pools in Azure Virtual Desktop, you're using a single-session scenario.

The following table provides examples for single-session ArcGIS Pro scenarios:

[] [Expand table](#)

Workload type	Example Azure virtual machine SKU	Activity
Light	NV4as_v4, NV8as_v4	Viewing
Medium	NV16as_v4, NC4as_T4_v3, NC8as_T4_v3, NV6ads_A10_v5	Editing
Heavy	NC16as_T4_v3, NV12ads_A10_v5, NV18ads_A10_v5	Visualizing

Multi-session recommendations

Multi-session scenarios are when there's more than one user signed in to a session host at any one time. For example, when you use pooled host pools in Azure Virtual Desktop with the Windows 11 Enterprise multi-session operating system (OS), that's a multi-session deployment.

The following table provides examples for multi-session ArcGIS Pro scenarios:

 [Expand table](#)

Workload type	Example Azure virtual machine SKU	Maximum users per VM	Activity
Light	NV18ads_A10_v5, NC16as_T4_v3, NV32as_v4	6	Viewing
Medium	NV18ads_A10_v5, NC16as_T4_v3, NV32as_v4	4	Editing
Heavy	NV18ads_A10_v5, NC16as_T4_v3, NV32as_v4	3	Visualizing

Contributors

This article is maintained by Microsoft. It was originally written by the following contributors.

Principal authors:

- [Matt Hallenborg](#) | (Senior Cloud Solution Architect)
- [Ron Vincent](#) | (Senior Program Manager)

To see non-public LinkedIn profiles, sign in to LinkedIn.

Next steps

- [Create a managed image of a generalized VM in Azure](#)
- [Prepare an Azure Virtual Desktop image with the Virtual Desktop Optimization Tool \(VDOT\)](#)
- [Download and install FSLogix](#)
- [Create a golden image in Azure](#)
- [Create an Azure Virtual Desktop host pool](#)
- [Create an Azure SQL Managed Instance](#)

- [Install ArcGIS Server ↗](#)
- [Install Portal for ArcGIS ↗](#)
- [Install NVIDIA GPU drivers on N-Series VMs running Windows](#)
- [Assess Azure SQL Managed Instance via SSMS ↗](#)
- [Configure public endpoint in Azure SQL Managed Instance](#)
- [Connect to Microsoft SQL Server from ArcGIS ↗](#)
- [Create Enterprise Geodatabase ↗](#)
- [Best practices for tuning ArcGIS Enterprise ↗](#)
- [Configure highly available ArcGIS Enterprise ↗](#)
- [Esri GIS mapping software, location intelligence, and spatial analytics ↗](#)

Related resources

- [Azure Virtual Desktop for the enterprise](#)
- [FSLogix configuration examples](#)

Web applications architecture design

07/11/2025

Many web apps are expected to be available all day, every day from anywhere in the world, and usable from virtually any device or screen size. Web applications must be secure, flexible, and scalable to meet spikes in demand.

This article provides an overview of Azure web app technologies, guidance, solution ideas, and reference architectures contained in the Azure Architecture Center.

Azure provides a wide range of tools and capabilities for creating, hosting, and monitoring web apps. These are just some of the key web app services available in Azure:

- [Azure App Service](#) enables you to easily create enterprise-ready web and mobile apps for any platform or device and deploy them on a scalable cloud infrastructure.
- [Azure Web Application Firewall](#) provides powerful protection for web apps.
- [Azure Monitor](#) provides full observability into your applications, infrastructure, and network. Monitor includes [Application Insights](#), which provides application performance management and monitoring for live web apps.
- [Azure SignalR Service](#) enables you to easily add real-time web functionalities.
- [Web App for Containers](#) enables you to run containerized web apps on Windows and Linux.
- [Azure Service Bus](#) enables you to integrate with other web apps using loosely coupled event-driven patterns.

Introduction to web apps on Azure

If you're new to creating and hosting web apps on Azure, the best way to learn more is with [Microsoft Learn training](#). This free online platform provides interactive training for Microsoft products and more.

These are a few good starting points to consider:

- [Create Azure App Service web apps](#)
- [Deploy and run a containerized web app with Azure App Service](#)

Path to production

Consider these patterns, guidelines, and architectures as you plan and implement your deployment:

- [Basic web application](#)
- [Baseline zone-redundant web application](#)
- [Common web application architectures](#)
- [Design principles for Azure applications](#)
- [Enterprise deployment using App Service Environment](#)
- [High availability enterprise deployment using App Service Environment](#)

Best practices

For a good overview, see [Characteristics of modern web applications](#).

For more information specific to Azure App Service, see:

- [Architecture best practices for Azure App Service \(Web Apps\)](#)
- [App Service deployment best practices](#)
- [Azure security baseline for App Service](#)

Web app architectures

The following sections, organized by category, provide links to sample web app architectures.

Modernization

- [Choose between traditional web apps and single-page apps](#)
- [ASP.NET architectural principles](#)
- [Common client-side web technologies](#)
- [Development process for Azure](#)
- [Azure hosting recommendations for ASP.NET Core web apps](#)

Multi-tier apps

- [Multi-tier web application built for HA/DR](#)

Scalability

- [Baseline web application with zone redundancy](#)

Security

- [Improved-security access to multitenant web apps from an on-premises network](#)

- Protect APIs with Application Gateway and API Management

SharePoint

- Highly available SharePoint farm

Stay current with web development

Get the latest [updates on Azure web app products and features](#).

Additional resources

Example solutions

Here are some additional implementations to consider:

- App Service networking features
- Migrate a web app using Azure APIM

AWS or Google Cloud professionals

- [AWS to Azure services comparison - Web applications](#)
- [Google Cloud to Azure services comparison - Application services](#)

Microsoft Cloud Adoption Framework

Proven guidance to be successful in Azure.



OVERVIEW
Cloud Adoption Framework
overview



HOW-TO GUIDE
AI adoption



ARCHITECTURE
Azure landing zone



GET STARTED
Adoption scenarios



GET STARTED
Create an Azure account ↗



GET STARTED
Claim startup credits ↗



HOW-TO GUIDE
Quick setup guide

Cloud Adoption Framework methodologies

Proven guidance for each phase of the cloud journey.

Strategy

Define business justification and Azure adoption outcomes.

Plan

Prepare people, processes, and technology for Azure adoption.

Ready

Migrate

Prepare your Azure environment and landing zone for workloads.

Migrate your workloads to Azure from on-premises and other clouds.

Modernize

Improve your existing workloads to better meet business needs.

Govern

Govern your entire Azure environment.

Cloud-native

Build new cloud-native workloads and features in Azure.

Secure

Secure your entire Azure environment.

Manage

Manage your entire Azure environment.

Workload design guidance

Use the Well-Architected Framework and Azure Architecture Center to build Azure solutions.

Well-Architected Framework

Design workloads the right way.

Azure Architecture Center

Build using a catalog of Azure architectures.

Find help

Microsoft has dedicated resources to support your Azure adoption journey.

Azure Accelerate

Accelerate your cloud and AI adoption with experts from Microsoft.

Why the Microsoft Cloud Adoption Framework for Azure is right for your business. <https://aka.ms/adopt> 