



# Introduction

5 minutes

Azure Storage accounts provide a wealth of security options that protect your cloud-based data. Azure services such as Blob storage, Files share, Table storage, and Data Lake Store all build on Azure Storage. Because of this foundation, the services benefit from the fine-grained security controls in Azure Storage.

Imagine the network administrator at Contoso is auditing the security of the assets within the domain. At the end of the audit, she needs to be satisfied that all the data stored in Azure strictly follows Contoso's security policy. As Contoso's primary data consultant, you'll help the network administrator understand how Azure Storage can help her meet Contoso's security requirements.

## Learning objectives

In this module you will:

- Investigate the ways Azure Storage protects your data
- Explore the authentication options to access data
- Learn about Advanced Threat Protection
- Learn how to control network access to data
- Explore the Azure Data Lake enterprise-class security features

---

**Next unit: Explore Azure Storage security features**

Continue >



# Explore Azure Storage security features

10 minutes

Contoso relies heavily on massive amounts of data in Azure Storage. Their many applications rely on blobs, unstructured table storage, Azure Data Lake, and Server Message Block (SMB)-based file shares.

After Contoso's competitor has a data breach, Contoso tasks the network administrator to check the organization's data security. As Contoso's data consultant, you assure the network administrator that Azure Storage accounts provide several high-level security benefits for the data in the cloud:

- Protect the data at rest
- Protect the data in transit
- Support browser cross-domain access
- Control who can access data
- Audit storage access

## Encryption at rest

All data written to Azure Storage is automatically encrypted by Storage Service Encryption (SSE) with a 256-bit Advanced Encryption Standard (AES) cipher. SSE automatically encrypts data when writing it to Azure Storage. When you read data from Azure Storage, Azure Storage decrypts the data before returning it. This process incurs no additional charges and doesn't degrade performance. It can't be disabled.

For virtual machines (VMs), Azure lets you encrypt virtual hard disks (VHDs) by using Azure Disk Encryption. This encryption uses BitLocker for Windows images, and it uses dm-crypt for Linux.

Azure Key Vault stores the keys automatically to help you control and manage the disk-encryption keys and secrets. So even if someone gets access to the VHD image and downloads it, they can't access the data on the VHD.

## Encryption in transit

Keep your data secure by enabling *transport-level security* between Azure and the client. Always use *HTTPS* to secure communication over the public internet. When you call the REST APIs to access objects in storage accounts, you can enforce the use of HTTPS by requiring [secure transfer](#) for the storage account. After you enable secure transfer, connections that use HTTP will be refused. This flag will also enforce secure transfer over SMB by requiring SMB 3.0 for all file share mounts.

## CORS support

Contoso stores several website asset types in Azure Storage. These types include images and videos. To secure browser apps, Contoso locks GET requests down to specific domains.

Azure Storage supports cross-domain access through cross-origin resource sharing (CORS). CORS uses HTTP headers so that a web application at one domain can access resources from a server at a different domain. By using CORS, web apps ensure that they load only authorized content from authorized sources.

CORS support is an optional flag you can enable on Storage accounts. The flag adds the appropriate headers when you use HTTP GET requests to retrieve resources from the Storage account.

## Role-based access control

To access data in a storage account, the client makes a request over HTTP or HTTPS. Every request to a secure resource must be authorized. The service ensures that the client has the permissions required to access the data. You can choose from several access options. Arguably, the most flexible option is role-based access.

Azure Storage supports Azure Active Directory and role-based access control (RBAC) for both resource management and data operations. To security principals, you can assign RBAC roles that are scoped to the storage account. Use Active Directory to authorize resource management operations, such as configuration. Active Directory is supported for data operations on Blob and Queue storage.

To a security principal or a managed identity for Azure resources, you can assign RBAC roles that are scoped to a subscription, a resource group, a storage account, or an individual container or queue.

## Auditing access

Auditing is another part of controlling access. You can audit Azure Storage access by using the built-in Storage Analytics service.

Storage Analytics logs every operation in real time, and you can search the Storage Analytics logs for specific requests. Filter based on the authentication mechanism, the success of the operation, or the resource that was accessed.

---

**Next unit: Understand storage account keys**

Continue >



# Understand storage account keys


5 minutes

Much of Contoso's data is generated or consumed by custom applications. The applications are written in various languages.


Azure Storage accounts can create authorized apps in Active Directory to control access to the data in blobs and queues. This authentication approach is the best solution for apps that use Blob storage or Queue storage.

For other storage models, clients can use a *shared key*, or shared secret. This authentication option is one of the easiest to use, and it supports blobs, files, queues, and tables. The client embeds the shared key in the HTTP `Authorization` header of every request, and the Storage account validates the key.

For example, an application can issue a `GET` request against a blob resource:

	 Copy
<code>GET http://myaccount.blob.core.windows.net/?restype=service&amp;comp=stats</code>	

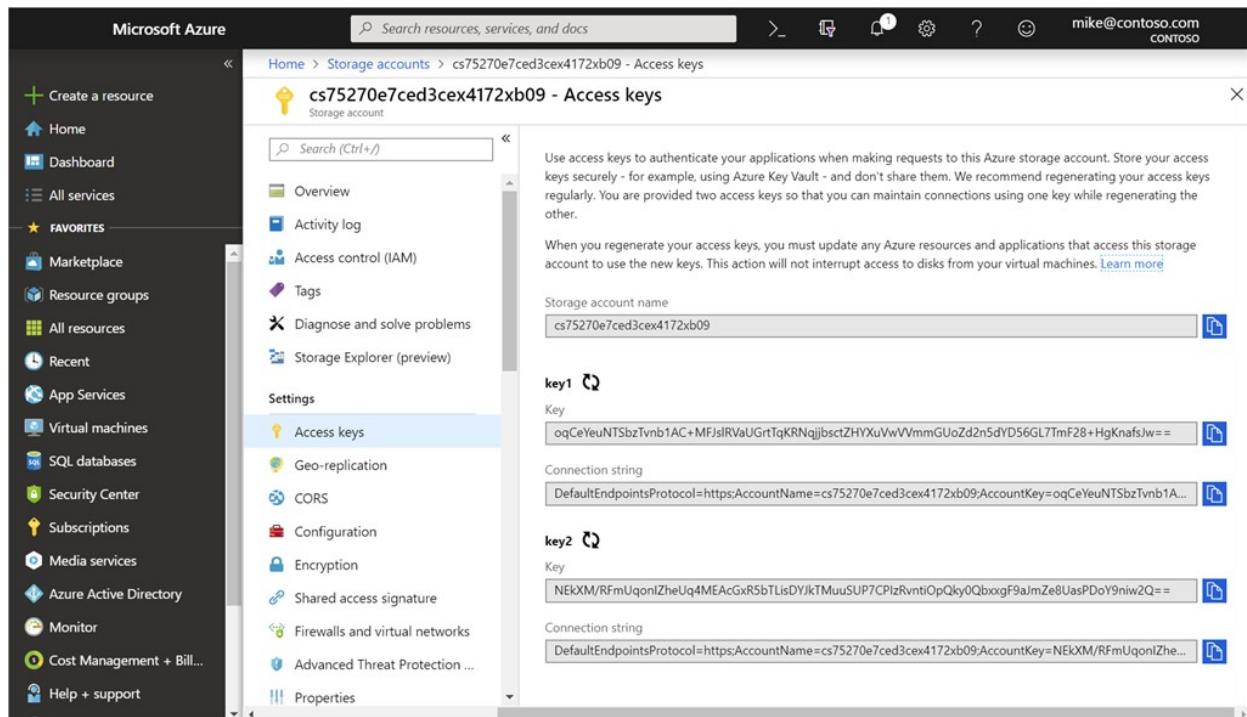
HTTP headers control the version of the REST API, the date, and the encoded shared key:

	 Copy
<code>x-ms-version: 2018-03-28 Date: Wed, 23 Oct 2018 21:00:44 GMT Authorization: SharedKey myaccount:CY10P303jGFpYFbTCBimLn0Xo-v0vt0khH/E5Gy0fXvg=</code>	

# Storage account keys

In Azure Storage accounts, shared keys are called *storage account keys*. Azure creates two of these keys (primary and secondary) for each storage account you create. The keys give access to *everything* in the account.

You'll find the storage account keys in the Azure portal view of the storage account. Just select **Settings** > **Access keys**.



## Protecting shared keys

The storage account has only two keys, and they provide full access to the account. Because these keys are powerful, use them only with trusted in-house applications that you control completely.

If the keys are compromised, change the key values in the Azure portal. Here are several reasons to regenerate your storage account keys:

- For security reasons, you might regenerate keys periodically.

- If someone hacks into an application and gets the key that was hard-coded or saved in a configuration file, regenerate the key. The compromised key can give the hacker full access to your storage account.
- If your team is using a Storage Explorer application that keeps the storage account key, and one of the team members leaves, regenerate the key. Otherwise, the application will continue to work, giving the former team member access to your storage account.

To refresh keys:

1. Change each trusted app to use the secondary key.
2. Refresh the primary key in the Azure portal. Consider this as the new secondary key value.

**ⓘ Important**

After your refresh keys, any client that attempts to use the old key value will be refused. Make sure you identify all clients that use the shared key, and update them to keep them operational.

---

**Next unit: Understand shared access signatures**

Continue >



# Understand shared access signatures

5 minutes

As a best practice, you shouldn't share storage account keys with external third-party applications. If these apps need access to your data, you'll need to secure their connections without using storage account keys.

For untrusted clients, use a *shared access signature* (SAS). A shared access signature is a string that contains a security token that can be attached to a URI. Use a shared access signature to delegate access to storage objects and specify constraints, such as the permissions and the time range of access.

You can give a customer a shared access signature token, for example, so they can upload pictures to a file system in Blob storage. Separately, you can give a web application permission to read those pictures. In both cases, you allow only the access that the application needs to do the task.

## Types of shared access signatures

You can use a *service-level* shared access signature to allow access to specific resources in a storage account. You'd use this type of shared access signature, for example, to allow an app to retrieve a list of files in a file system or to download a file.

Use an *account-level* shared access signature to allow access to anything that a service-level shared access signature can allow, plus additional resources and abilities. For example, you can use an account-level shared access signature to allow the ability to create file systems.

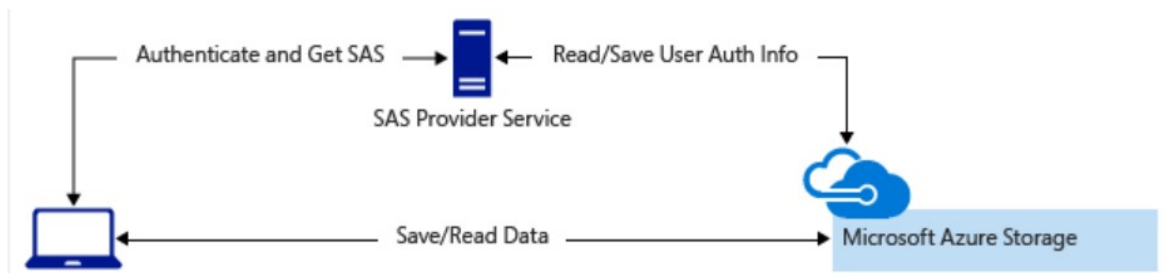
You'd typically use a shared access signature for a service where users read and write their data to your storage account. Accounts that store user data have two typical designs:



- Clients upload and download data through a front-end proxy service, which performs authentication. This front-end proxy service has the advantage of allowing validation of business rules. But if the service must handle large amounts of data or high-volume transactions, you might find it complicated or expensive to scale this service to match demand.



- A lightweight service authenticates the client as needed. Then it generates a shared access signature. After receiving the shared access signature, the client can access storage account resources directly. The shared access signature defines the client's permissions and access interval. The shared access signature reduces the need to route all data through the front-end proxy service.



**Next unit: Control network access to your storage account**

[Continue >](#)



# Control network access to your storage account

5 minutes

By default, storage accounts accept connections from clients on any network. To limit access to selected networks, you must first change the default action. You can restrict access to specific IP addresses, ranges, or virtual networks.

## Important

Changing network rules can affect your application's ability to connect to Azure Storage. If you set the default network rule to *deny*, you'll block all access to the data unless specific network rules *grant* access. Before you change the default rule to deny access, be sure to use network rules to grant access to any allowed networks.

## Manage default network access rules

Manage default network access rules for storage accounts through the Azure portal, PowerShell, or the Azure CLI.

To change default network access in the Azure portal:

1. Go to the storage account you want to secure.
2. Select **Firewalls and virtual networks**.
3. To restrict traffic from selected networks, choose **Selected networks**. To allow traffic from all networks, select **All networks**.
4. Select **Save** to apply your changes.

cs75270e7ced3cex4172xb09 - Firewalls and virtual networks

Storage account

Search (Ctrl+)

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Storage Explorer (preview)

Settings

Access keys

Geo-replication

CORS

Configuration

Encryption

Shared access signature

Firewalls and virtual networks

Advanced Threat Protection ...

Properties

Save Discard Refresh

Firewall settings allowing access to storage services will remain in effect for up to a minute after saving updated settings restricting access.

Allow access from

All networks

Selected networks

Configure network security for your storage accounts. [Learn more.](#)

Virtual networks

Secure your storage account with virtual networks. [+ Add existing virtual network](#) [+ Add new virtual network](#)

VIRTUAL NETWO...	SUBNET	ADDRESS RANGE	ENDPOINT STAT...	RESOURCE GROUP	SUBSCRIPTION
No network selected.					

Firewall

Add IP ranges to allow access from the internet or your on-premises networks. [Learn more.](#)

Add your client IP address ('90.50.100.200')

ADDRESS RANGE

IP address or CIDR

Exceptions

Allow trusted Microsoft services to access this storage account

Next unit: Understand Advanced Threat Protection for Azure Storage

Continue >



# Understand Advanced Threat Protection for Azure Storage

5 minutes

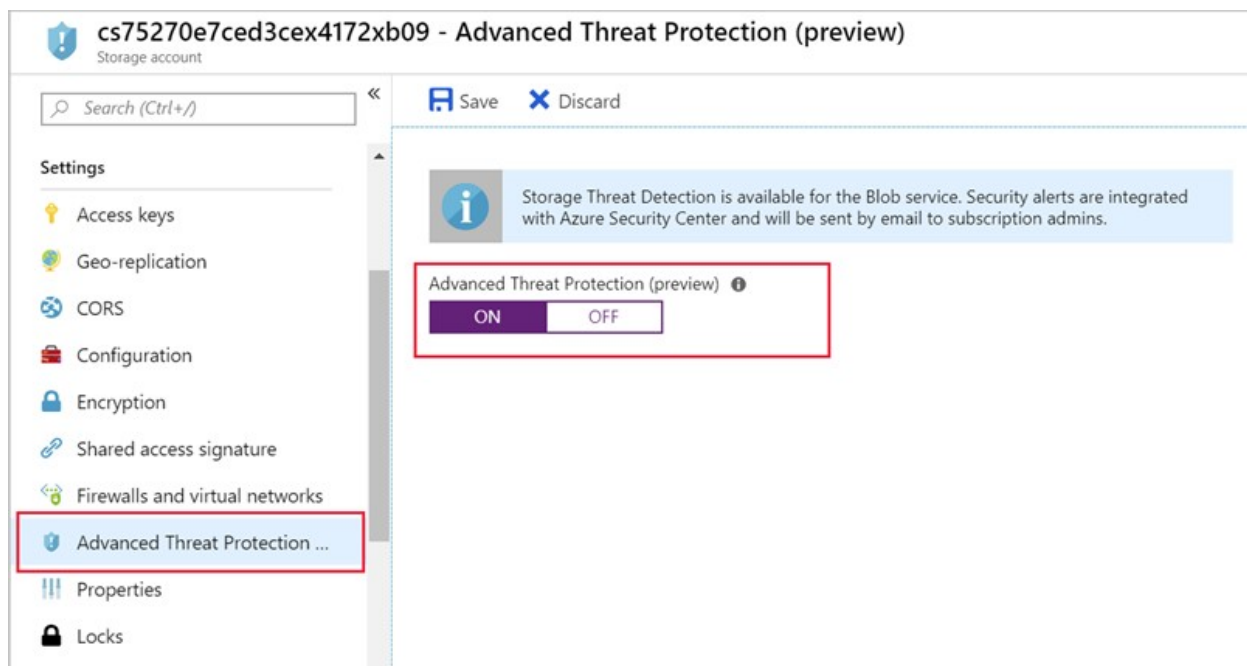
Detecting threats to your data is an important part of security. You can check an audit trail for all activity against a storage account. But that will often only show you that an intrusion *has already occurred*. What you really want is a way to be notified when suspicious activity is happening. That's where the Advanced Threat Protection feature in Azure Storage can help.

Advanced Threat Protection, now in public preview, detects anomalies in account activity. It then notifies you of potentially harmful attempts to access your account. You don't have to be a security expert or manage security monitoring systems to take advantage of this layer of threat protection.

Currently, Advanced Threat Protection for Azure Storage is available for the Blob service. Security alerts are integrated with Azure Security Center. The alerts are sent by email to subscription admins.

In the Azure portal, you can turn on threat protection on the configuration page of the Azure Storage account.

1. On the **Settings** page, select **Advanced Threat Protection**.
2. Turn **ON** Advanced Threat Protection.
3. Select **Save** to save the Advanced Threat Protection policy.



Next unit: Explore Azure Data Lake Storage security features

Continue >



# Explore Azure Data Lake Storage security features

5 minutes

Azure Data Lake Storage Gen2 provides a first-class data lake solution that allows enterprises to pull together their data. It's built on Azure Blob storage, so it inherits all of the security features we've reviewed in this module.

Along with role-based access control (RBAC), Azure Data Lake Storage Gen2 provides access control lists (ACLs) that are POSIX-compliant and that restrict access to only authorized users, groups, or service principals. It applies restrictions in a way that's flexible, fine-grained, and manageable. Azure Data Lake Storage Gen2 authenticates through Azure Active Directory OAuth 2.0 bearer tokens. This allows for flexible authentication schemes, including federation with Azure AD Connect and multifactor authentication that provides stronger protection than just passwords.

More significantly, these authentication schemes are integrated into the main analytics services that use the data. These services include Azure Databricks, HDInsight, and SQL Data Warehouse. Management tools such as Azure Storage Explorer are also included. After authentication finishes, permissions are applied at the finest granularity to ensure the right level of authorization for an enterprise's big-data assets.

The Azure Storage end-to-end encryption of data and transport layer protections complete the security shield for an enterprise data lake. The same set of analytics engines and tools can take advantage of these additional layers of protection, resulting in complete protection of your analytics pipelines.

---

**Next unit: Summary**

Continue >





# Summary

5 minutes

Azure Storage provides a layered security model. Use this model to secure your storage accounts to a specific set of supported networks. When you set up network rules, only applications that request data over the specified networks can access your storage account.

Authorization is supported by a public preview of Azure Active Directory credentials (for blobs and queues), a valid account access key, or a shared access signature (SAS) token. Data encryption is enabled by default, and you can proactively monitor systems by using Advanced Threat Protection.

## Check your knowledge

1. You are working on a project with a 3rd party vendor to build a website for a customer. The image assets that will be used on the website are stored in an Azure Storage account that is held in your subscription. You want to give read access to this data for a limited period of time. What security option would be the best option to use?

☐ CORS Support

☐ Storage Account

☒ Shared Access Signatures



**Correct. A shared access signature is a string that contains a security token that can be attached to a URI. Use a shared access signature to delegate access to storage objects and specify constraints, such as the permissions and the time range of access.**

2. When configuring network access to your Azure Storage Account, what is the default network rule?



To allow all connections from all networks

**Correct. The default network rule is to allow all connections from all networks.**

- ☐ To allow all connection from a private IP address range
- ☐ To deny all connections from all networks

3. Which Azure service detects anomalies in account activities and notifies you of potential harmful attempts to access your account?

☒ Advanced Threat Protection



**Advanced Threat Protection, detects anomalies in account activity. It then notifies you of potentially harmful attempts to access your account.**

- ☐ Azure Storage Account Security Feature
- ☐ Encryption in transit

---

**Module complete:**

Unlock achievement