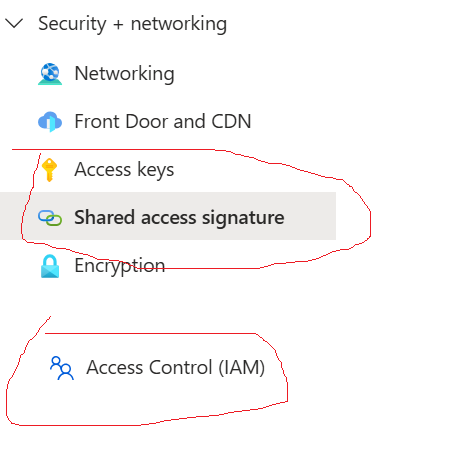
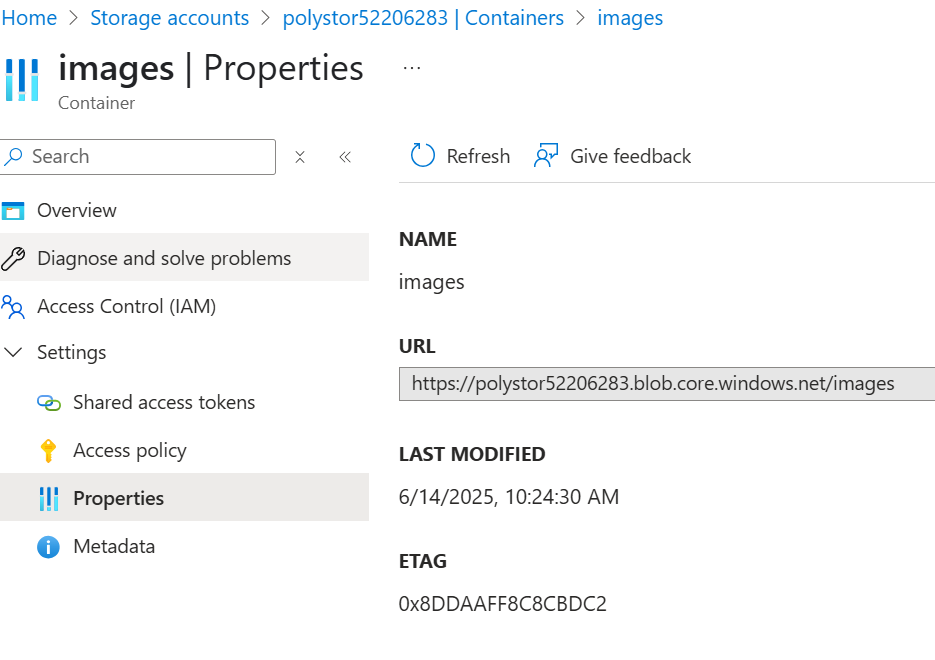
Blob Storage Security

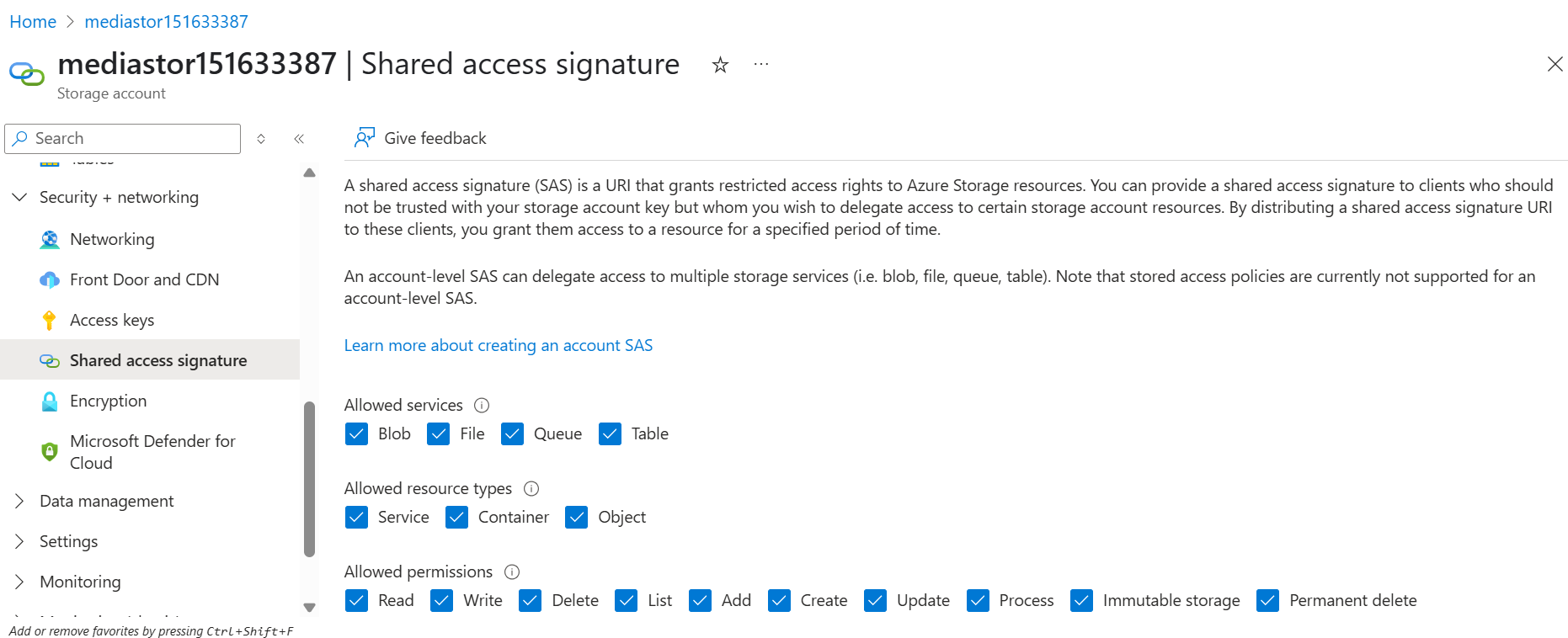
Storage Level security

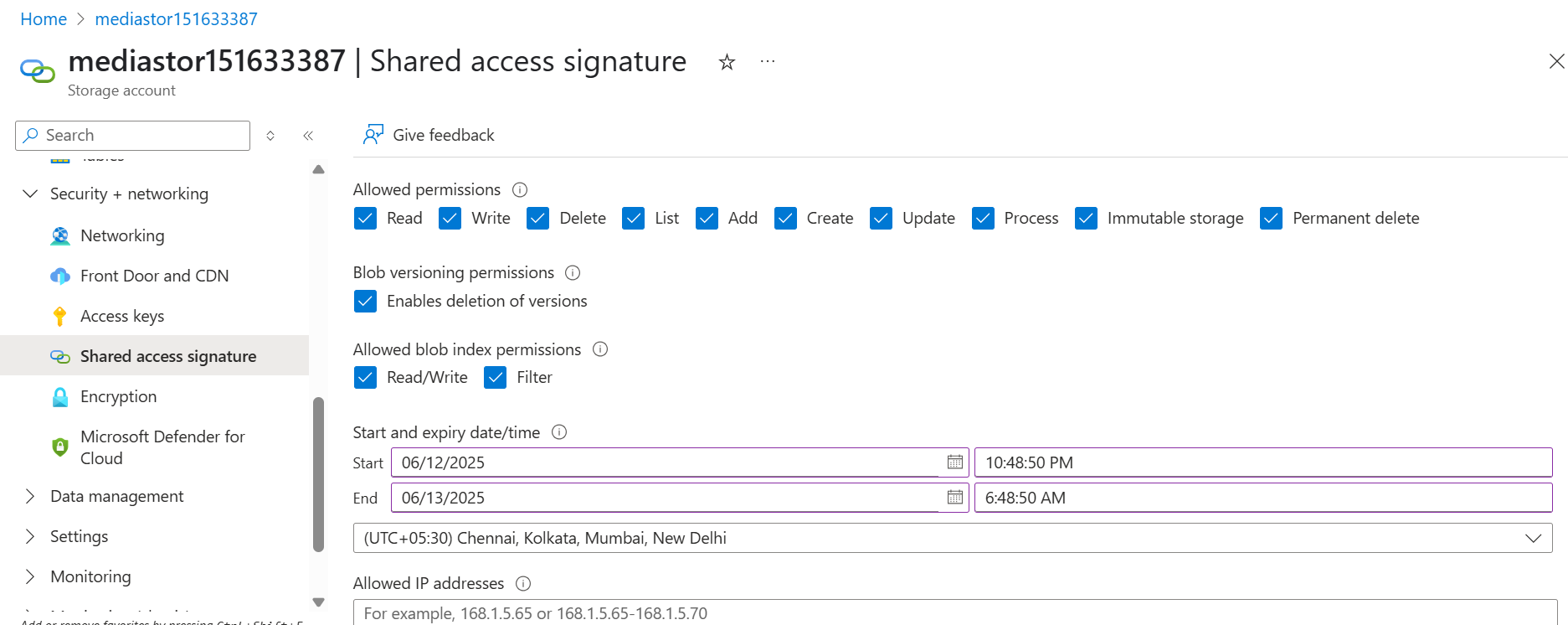


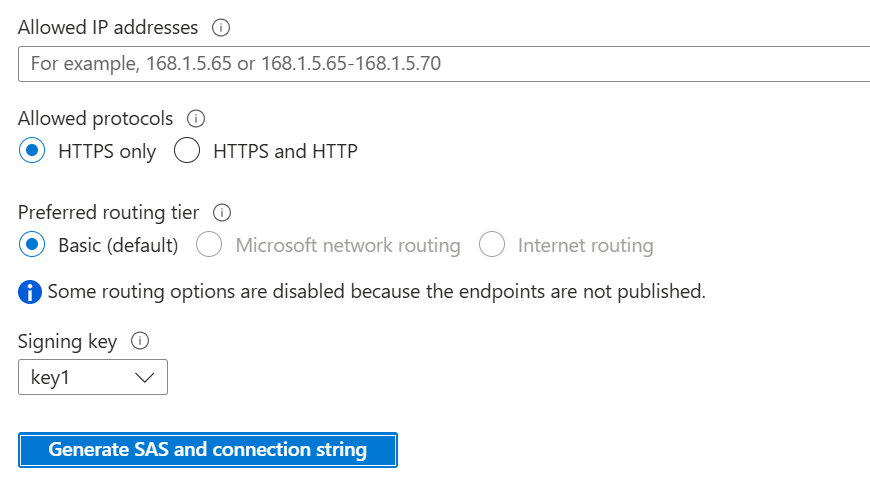
Container level security



1. SAS











Connection string

BlobEndpoint=https://mediastor151633387.blob.core.windows.net/;QueueEndpoint=https://mediastor151633387.queue.core.windows.net/;FileEndpoint=https://mediastor151633387.file.core.windows.net/;TableEndpoint=https://mediastor151633387.table.core.windows.net/;SharedAccessSignature=sv=2024-11-04&ss=bfqt&srt=sco&sp=rwdlacupiytfx&se=2025-06-13T01:18:50Z&st=2025-06-12T17:18:50Z&spr=https&sig=kFOo0Y9Uwf6SqqWlLFdc7Op9i5bZgGdvoeo3Dvff%2FYU%3D

SAS token

<https://mediastor151633387.blob.core.windows.net/?sv=2024-11-04&ss=bfqt&srt=sco&sp=rwdlacupiytfx&se=2025-06-13T01:18:50Z&st=2025-06-12T17:18:50Z&spr=https&sig=kFOo0Y9Uwf6SqqWlLFdc7Op9i5bZgGdvoeo3Dvff%2FYU%3D>

Blob service SAS URL

<https://mediastor151633387.blob.core.windows.net/?sv=2024-11-04&ss=bfqt&srt=sco&sp=rwdlacupiytfx&se=2025-06-13T01:18:50Z&st=2025-06-12T17:18:50Z&spr=https&sig=kFOo0Y9Uwf6SqqWlLFdc7Op9i5bZgGdvoeo3Dvff%2FYU%3D>

1. Access keys

Storage account name

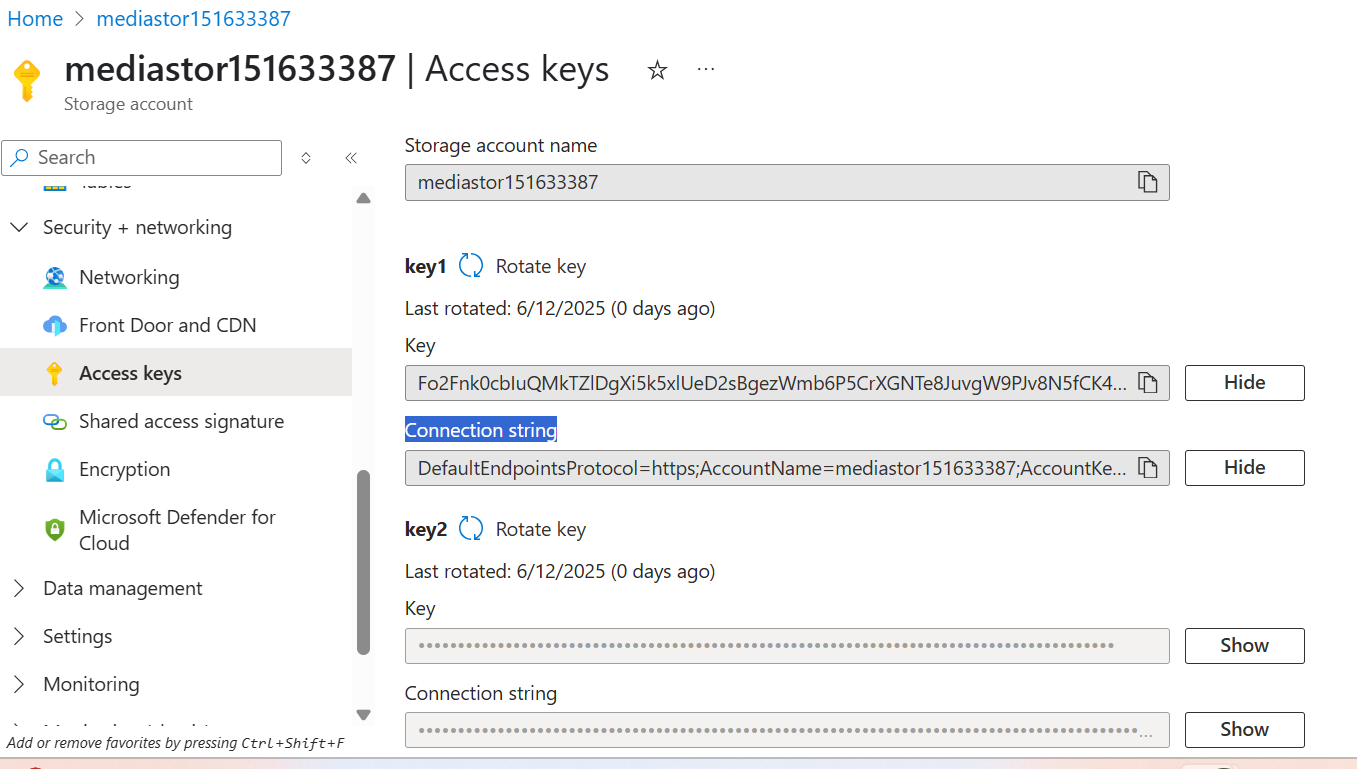
mediastor151633387

Key

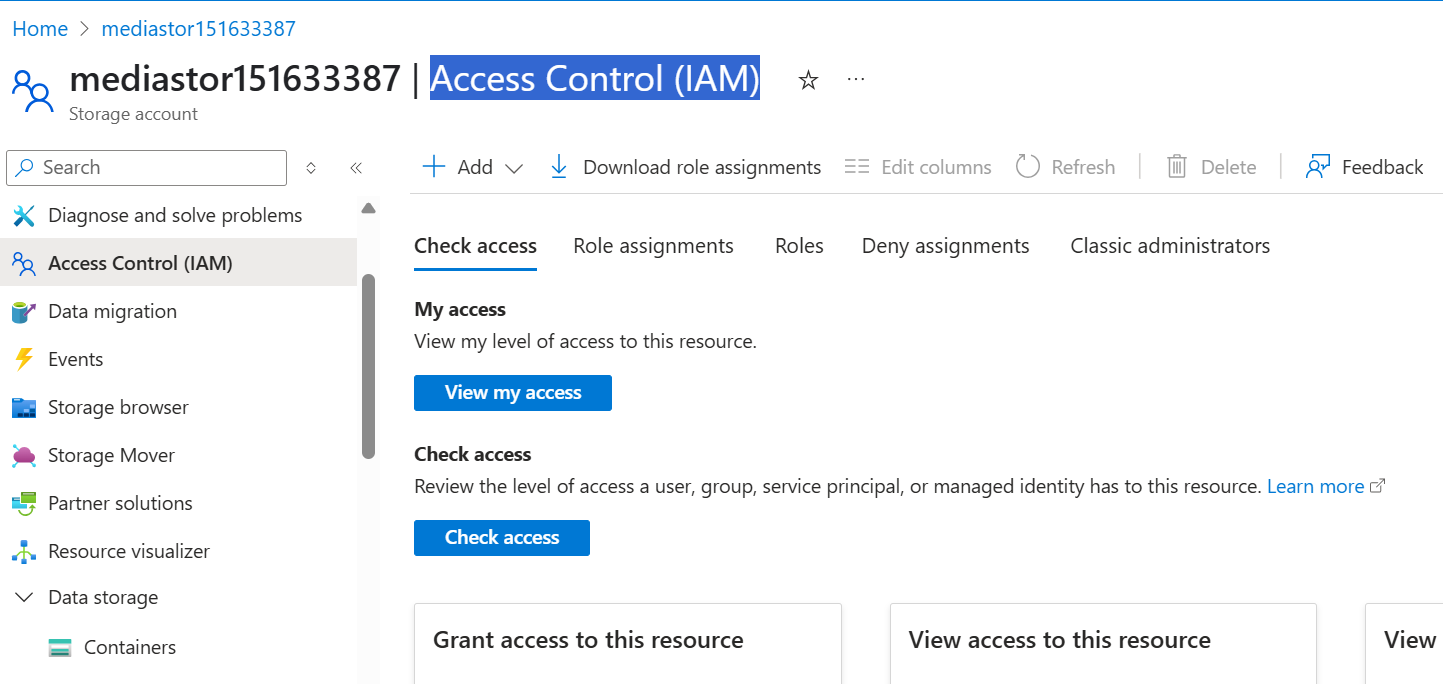
Fo2Fnk0cbIuQMkTZlDgXi5k5xlUeD2sBgezWmb6P5CrXGNTe8JuvgW9PJv8N5fCK4YEzGiRt2KId+ASt/rcZjQ==

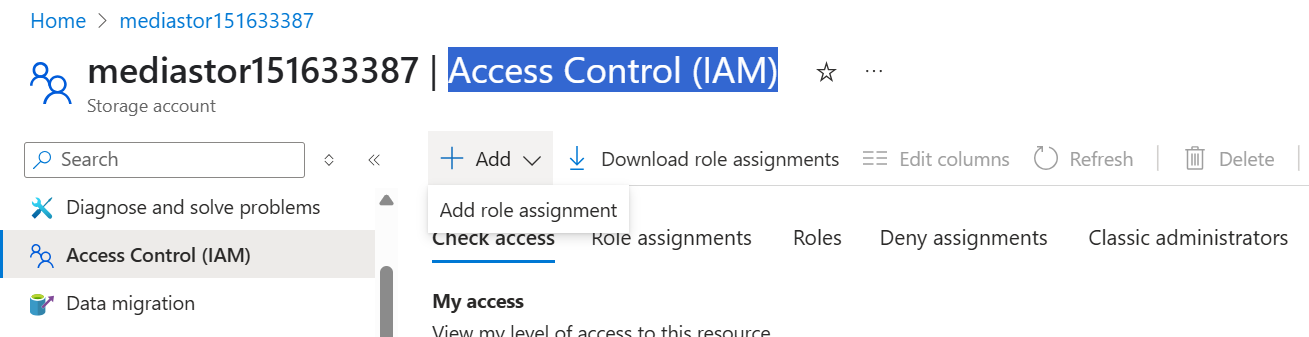
Connection string

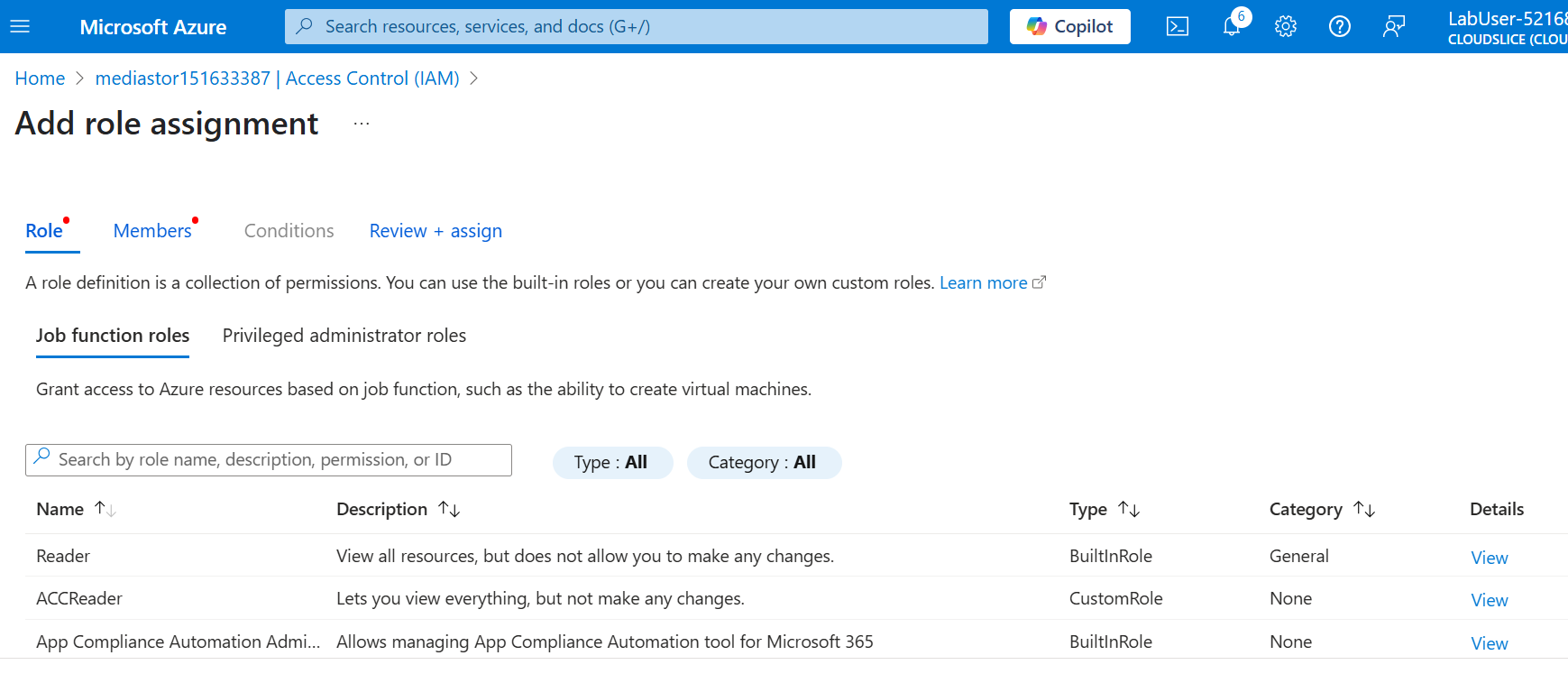
DefaultEndpointsProtocol=https;AccountName=mediastor151633387;AccountKey=Fo2Fnk0cbIuQMkTZlDgXi5k5xlUeD2sBgezWmb6P5CrXGNTe8JuvgW9PJv8N5fCK4YEzGiRt2KId+ASt/rcZjQ==;EndpointSuffix=core.windows.net

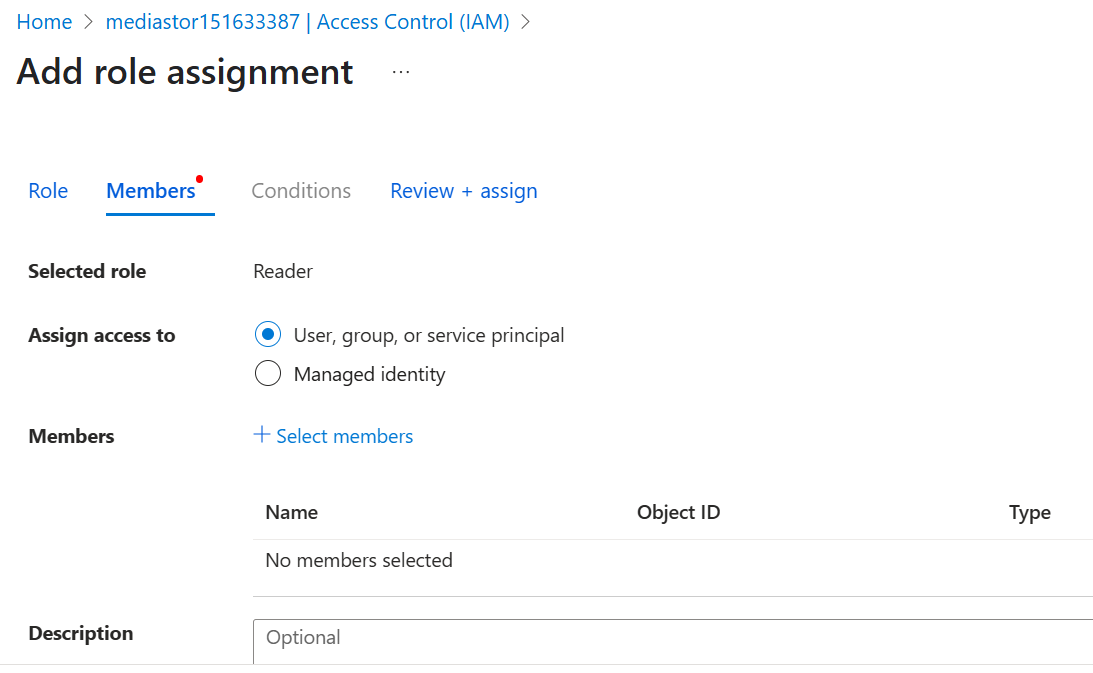


1. Access Control (IAM)

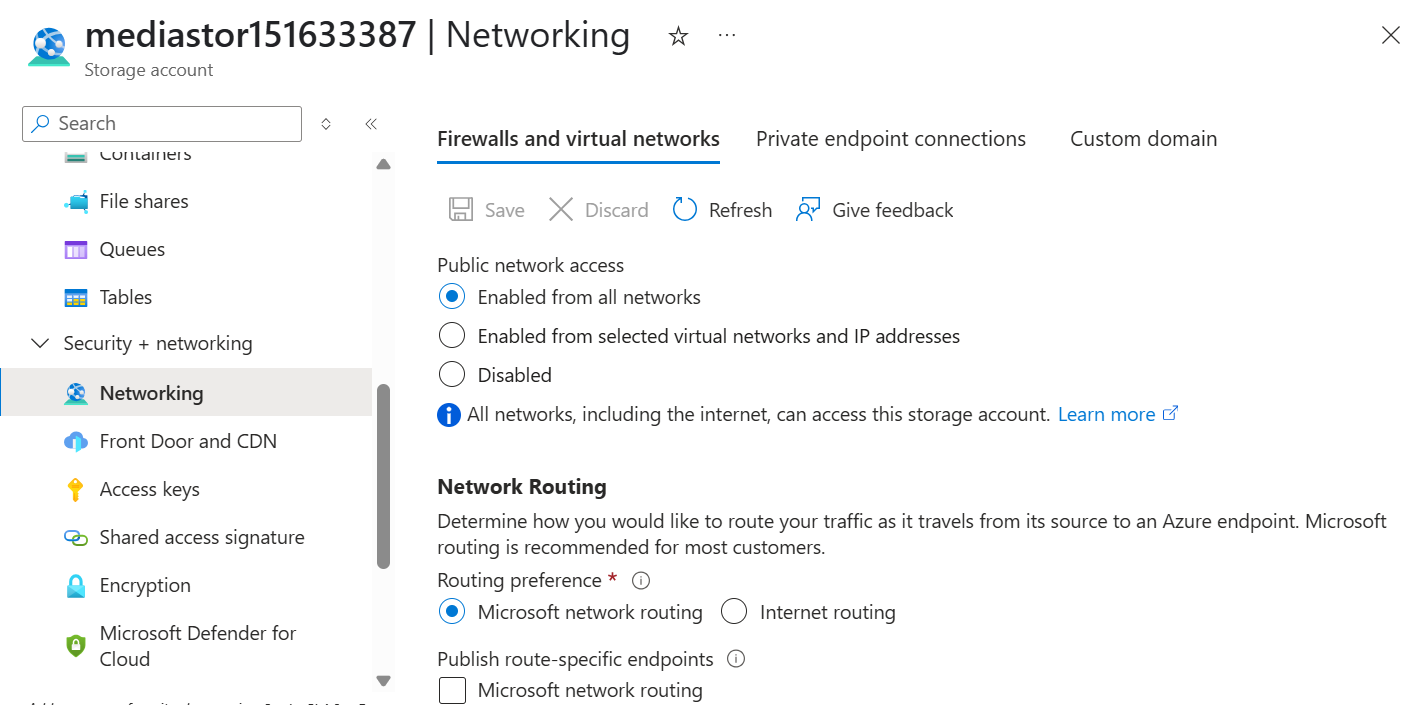


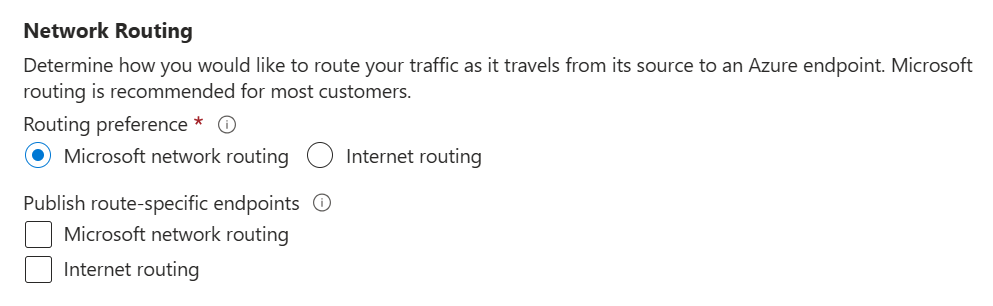


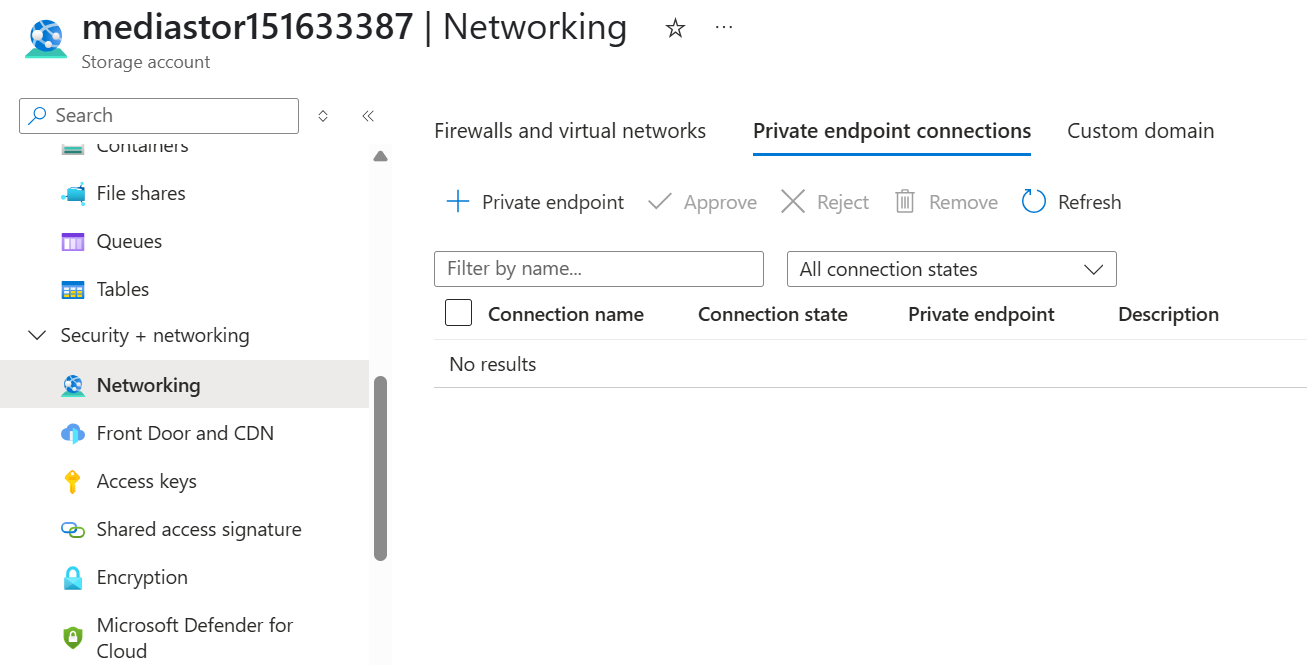




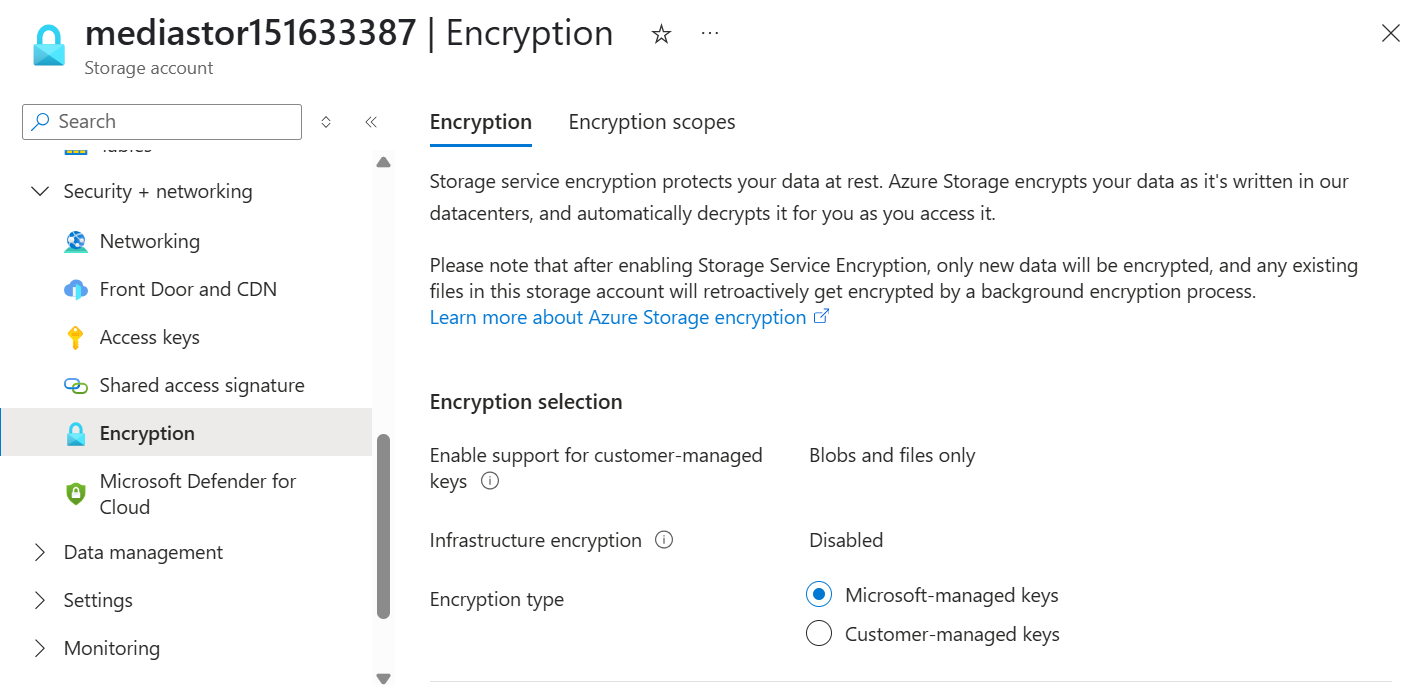
1. Networking

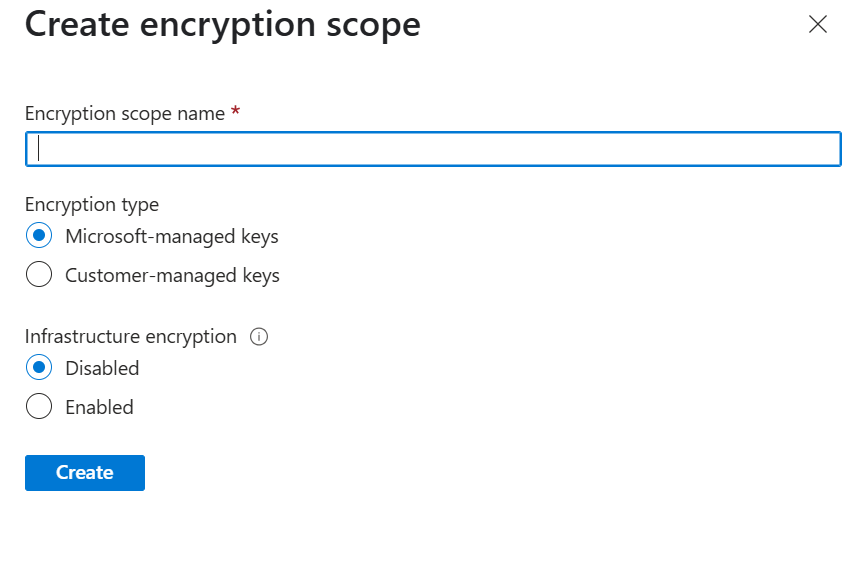






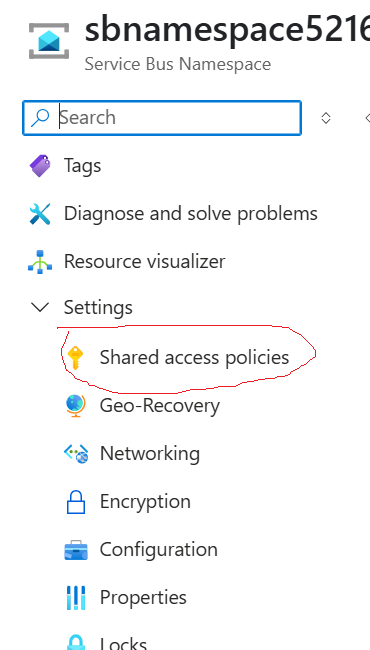
1. Encryption (By default Microsoft managed keys enabled automatically)



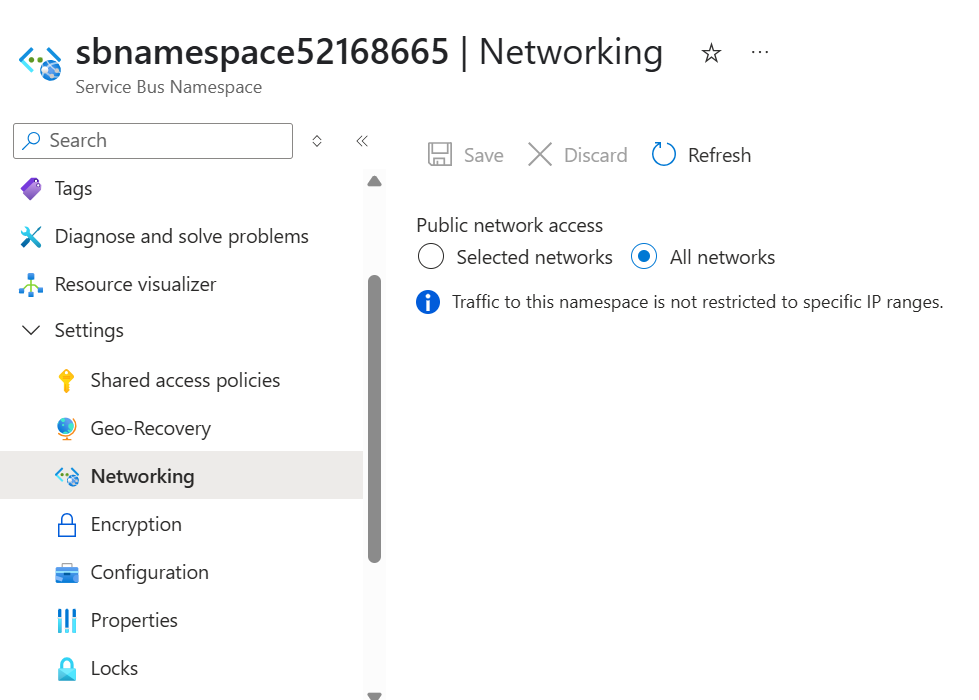


Secure Service Bus

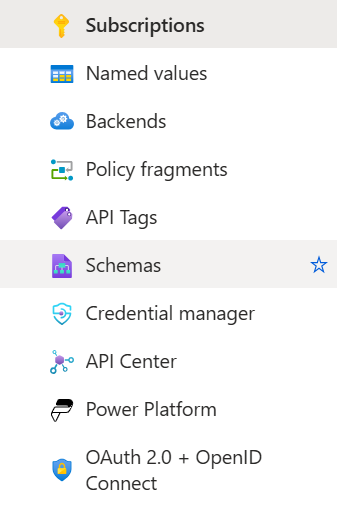
1. Shared access policies



1. Networking

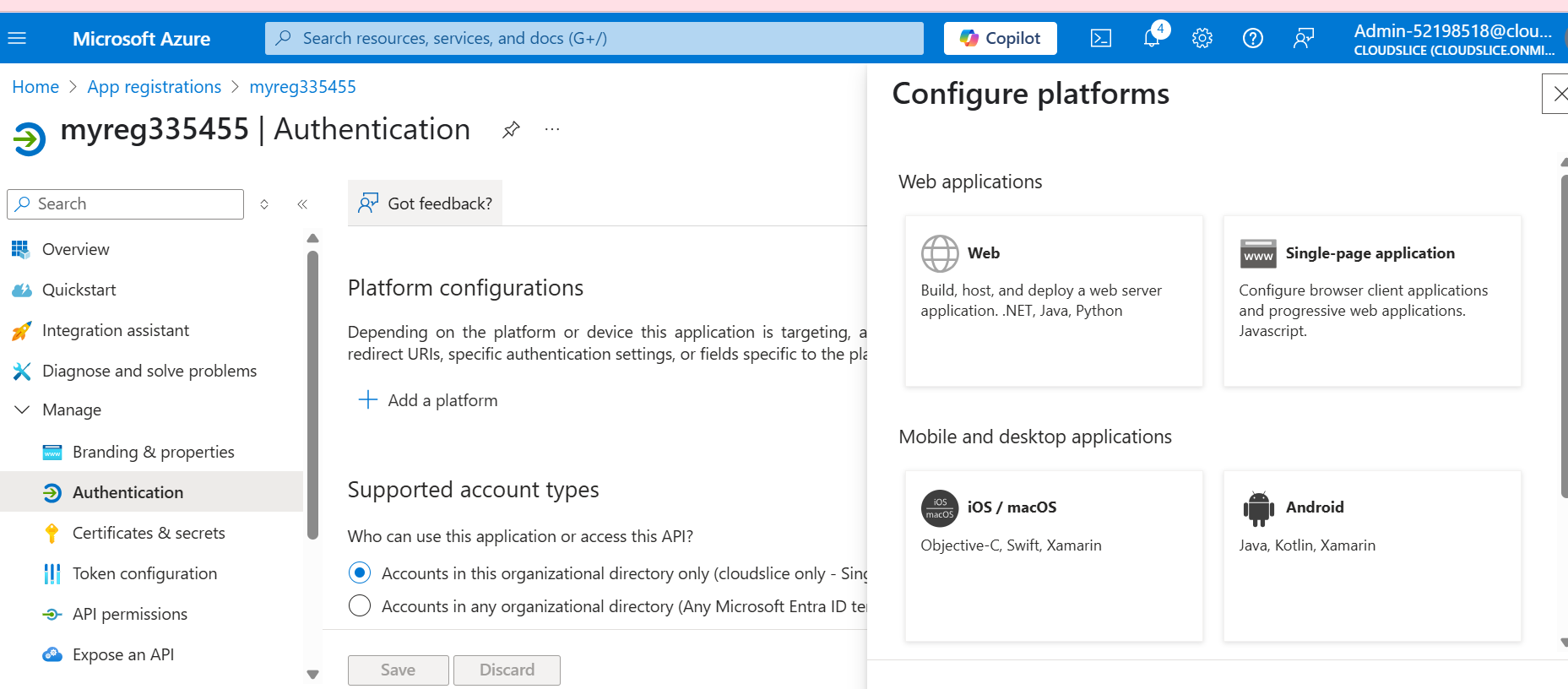


Secure APIM



1. Subscription
2. Oauth 2.0+OpenId Connect
3. Credential manager

| **Mechanism** | **Description** | **Considerations** |
| --- | --- | --- |
| [mTLS](https://learn.microsoft.com/en-us/azure/api-management/api-management-howto-mutual-certificates-for-clients) | [Validate certificate](https://learn.microsoft.com/en-us/azure/api-management/validate-client-certificate-policy) presented by the connecting client and check certificate properties against a certificate managed in API Management | Certificate may be stored in a key vault. |
| [Restrict caller IPs](https://learn.microsoft.com/en-us/azure/api-management/ip-filter-policy) | Filter (allow/deny) calls from specific IP addresses or address ranges. | Use to restrict access to certain users or organizations, or to traffic from upstream services. |
| [Subscription key](https://learn.microsoft.com/en-us/azure/api-management/api-management-subscriptions) | Limit access to one or more APIs based on an API Management [subscription](https://learn.microsoft.com/en-us/azure/api-management/api-management-howto-create-subscriptions) | We recommend using a subscription (API) key *in addition to* another method of authentication or authorization. On its own, a subscription key isn't a strong form of authentication, but use of the subscription key might be useful in certain scenarios, for example, tracking individual customers' API usage or granting access to specific API products. |

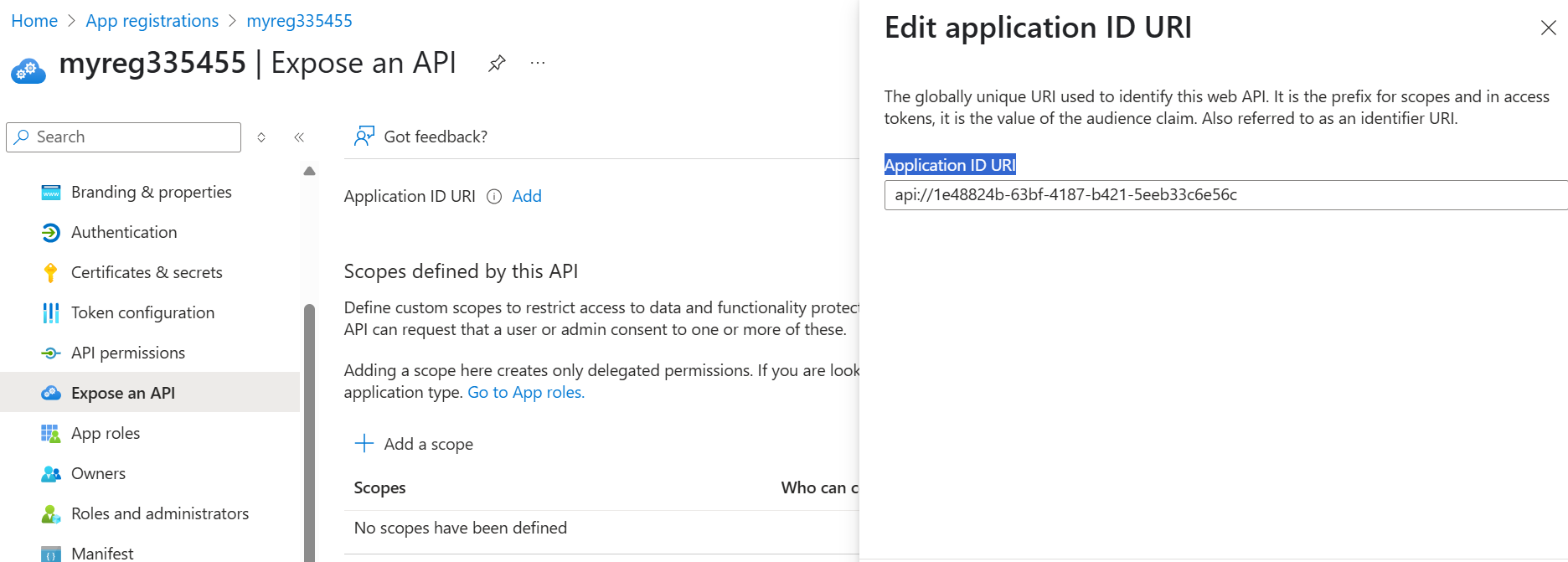


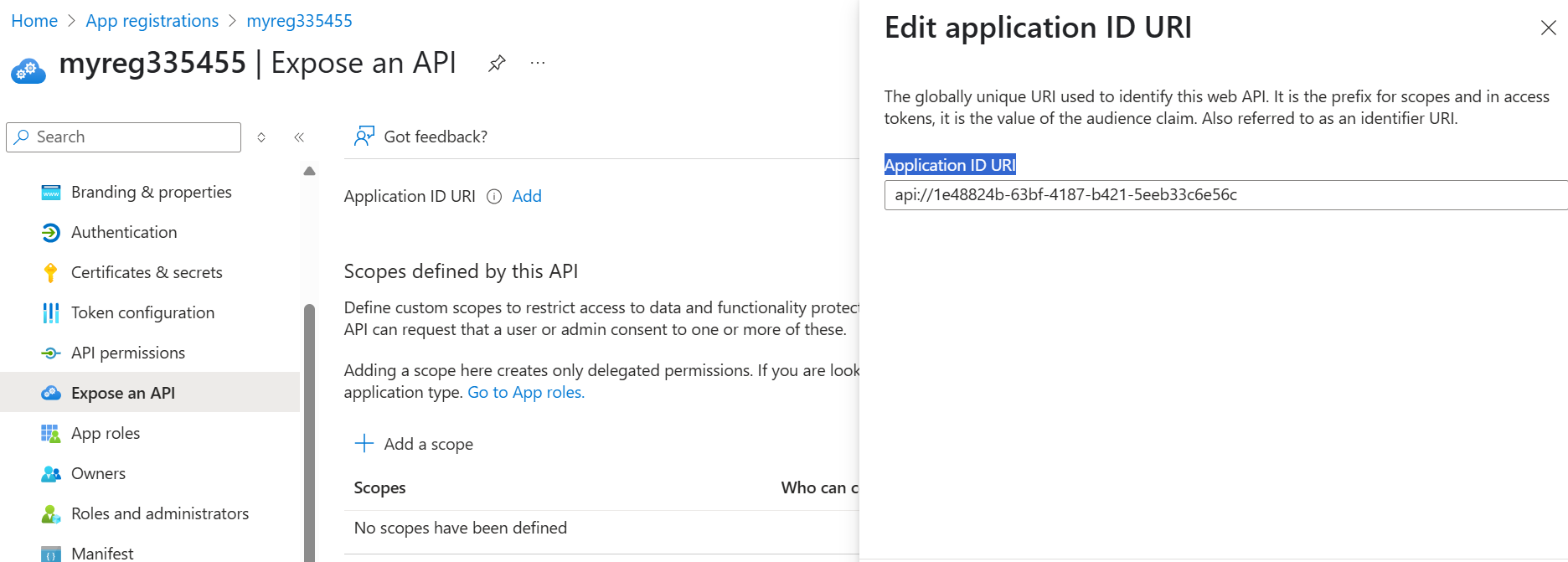
Secret

elS8Q~YuUympq6LV3xXC1p45g5sFJga1qlWGidq\_

Application ID URI

api://1e48824b-63bf-4187-b421-5eeb33c6e56c





<validate-jwt header-name=”Authorization” failed-validation-httpcode=”401" failed-validation-error-message=”Unauthorized. Access token is missing or invalid.”>

<openid-config url=”https://login.microsoftonline.com/\_\_tenandId\_\_/.well-known/openid-configuration" />

<required-claims>

<claim name=”aud” match=”any”>

<value>\_\_AppIdUri\_\_</value>

<value>\_\_apiProxyClientId\_\_</value>

</claim>

<claim name=”appid” match=”any”>

<value>\_\_apiClientClientId\_\_</value>

</claim>

</required-claims>

</validate-jwt>

<https://medium.com/@Subodh-Jain/azure-apim-jwt-inbound-policy-implementation-2068e5e36c2c>

<https://learn.microsoft.com/en-us/azure/api-management/api-management-howto-protect-backend-with-aad>

<validate-jwt header-name="Authorization" failed-validation-httpcode="401" failed-validation-error-message="Unauthorized. Access token is missing or invalid.">

<openid-config url="https://login.microsoftonline.com/8eb87a6e-8055-4135-b69d-f19c799ec045/v2.0/.well-known/openid-configuration" />

<audiences>

<audience>{audience-value - api://1e48824b-63bf-4187-b421-5eeb33c6e56c}</audience>

</audiences>

<issuers>

<issuer>{issuer-value - https://sts.windows.net/8eb87a6e-8055-4135-b69d-f19c799ec045/}</issuer>

</issuers>

<required-claims>

<claim name="aud">

<value>1e48824b-63bf-4187-b421-5eeb33c6e56c</value>

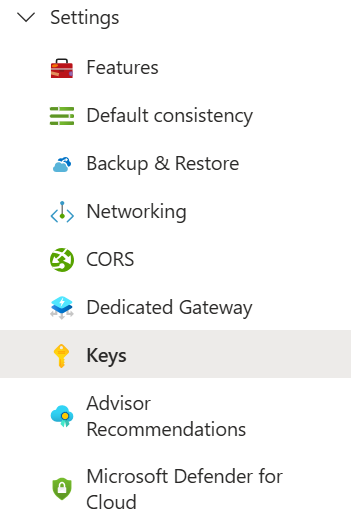
</claim>

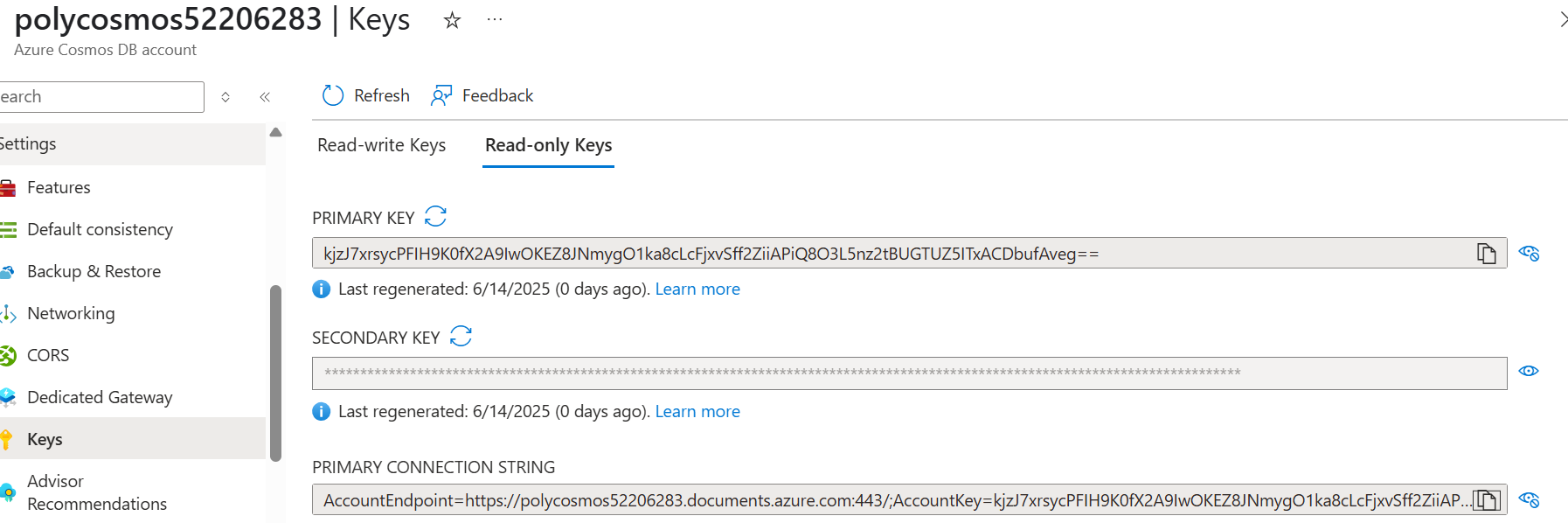
</required-claims>

</validate-jwt>

<https://medium.com/devops-dudes/getting-started-with-azure-apim-and-jwt-token-verification-ddf2ef68966b>

Cosmos DB





There are three scopes for role assignments:

* Account level (/)
* Database level (/dbs/<database-name>)
* Container level (/dbs/<database-name>/colls/<container-name>)

<https://azure.github.io/cloud-scale-data-for-devs-guide/security.html>

1. Key Vault