Security Best Practices for Azure Databricks

Version 2.0 – July 2024

Table of Contents

1. Introduction

Azure Databricks has worked with thousands of customers to securely deploy the Azure Databricks Data Intelligence Platform with the appropriate features to meet their security, privacy and regulatory requirements. While many organizations deploy security differently, there are patterns and features that are commonly used by most organizations.

Please note: unless you are a security specialist, there should be no need to read this entire document. You can implement our security best practices by following the **Define**, **Deploy**, **Monitor** approach outlined below:

- **Define:** Review the security checklists provided for most deployments and highly secure deployments below.
- **Deploy:** Our Security Reference Architecture (SRA) Terraform templates make it easy to deploy Azure Databricks workspaces that follow these best practices!
- **Monitor:** Use the Security Analysis Tool (SAT) for ongoing monitoring of adherence to security best practices.

> This document will focus on data platform security best practices, regardless of the types of workloads that you are running. For a comprehensive overview of security best practices relating to AI workloads, please refer to the Azure Databricks AI Security Framework (DASF).

2. Azure Databricks architecture

The Azure Databricks Data Intelligence Platform architecture is split into two separate planes to simplify your permissions, avoid data duplication and reduce risk. The control plane is the management plane where Azure Databricks runs the workspace application and manages notebooks, configuration and clusters. The compute plane handles your data processing. With serverless deployments, the compute plane exists in your Azure Databricks account rather than your cloud service provider account.

If you're new to the Azure Databricks platform, start with an overview of the architecture and a review of common security questions before you hop into specific recommendations. You'll see those at our Security and Trust Center, specifically the architecture overview.

3. Typical security configurations

Below, you will find the typical security configurations used by most customers. For simplicity, we've separated these into "most deployments" and "highly-secure deployments." Most deployments are as they sound – configurations that Azure Databricks expects to be present in most production or enterprise deployments such as the use of multi-factor authentication (MFA). Configurations for highly-secure deployments are more representative of what might be seen in environments with particularly sensitive data, intellectual property, or in regulated industries such as Healthcare, Life Sciences, or Financial Services, such as the use of Private Link connectivity.

Importantly, the recommendations outlined below are based on the types of configurations we see from our customers, who have different levels of risk tolerance. Because of this, and because every deployment is unique, the recommendations below are non-exhaustive and following them cannot guarantee that your deployment will be secure. Please review in the context of your overall enterprise security framework to determine what is required to secure your deployment and your data.

Most deployments

The following configurations are part of many production Azure Databricks deployments. If you are a small data science team working with data that is not particularly sensitive, you may not feel the need to deploy all of these. If instead you are analyzing large volumes of sensitive data, we recommend that you review these configurations more closely.

- ☐ Leverage multi-factor authentication for all user access
- ☐ Use Secure Cluster Connectivity (No Public IP)
- ☐ Deploy Azure Databricks into your own Azure virtual network for increased control over the network environment. Even if you do not need this now, this option increases your chances for future success with your initial workspace(s)
- ☐ Restrict access to your account, workspaces and Delta shares using IP access lists
- ☐ Use Unity Catalog for centralized data governance
- ☐ Use Azure Managed Identities to access storage
- ☐ Plan your workspace and data isolation models
- ☐ Consider whether to isolate Azure Databricks workspaces into different networks
- ☐ Prevent anonymous read access & apply other protections as per the Azure security baseline for Storage
- ☐ Consider whether your datasets require soft deletes and other data protection features
- ☐ Configure Azure Storage firewalls for storage accounts that store sensitive data
- ☐ Manage your code with Git folders and CI/CD
- ☐ Limit the number of admin users, enforce segregation of duties between regular and admin accounts and restrict workspace admins
- ☐ Run administrative tasks and production workloads with service principals
- ☐ Manage access according to the principle of least privilege
- ☐ Use compute that supports user isolation
- ☐ Configure and monitor system tables
- ☐ Control & monitor workspace access for Azure Databricks personnel
- ☐ Use OAuth or Azure Entra ID tokens and disable or restrict the use of Personal Access Tokens using token management
- ☐ Avoid storing production datasets in DBFS
- ☐ Store and use secrets securely
- ☐ Consider whether to implement network controls for data exfiltration protection
- ☐ Restart clusters on a regular schedule so that the latest patches are applied
- ☐ Use Delta Sharing & configure recipient token lifetimes for every metastore
- ☐ Implement a cost monitoring and charge-back strategy via budgets and tagging

Highly secure deployments

In addition to the configurations typical to most deployments, the following configurations are often used in highly-secure Azure Databricks deployments. While these are common configurations, not all highly

secure environments use all of these settings. We recommend incorporating appropriate items into your existing security practices, where informed by the threat models in the following section and your company's risk tolerance.

- ☐ Keep users and groups up-to-date using SCIM
- ☐ Use back-end PrivateLink
- ☐ Consider front-end PrivateLink
- ☐ Implement network controls for data exfiltration protection
- ☐ Isolate Azure Databricks workspaces into different networks
- ☐ Configure Azure Storage firewalls for all storage accounts
- ☐ Evaluate whether customer-managed encryption keys (for both managed services and storage) are needed for increased control over data at rest
- ☐ Consider whether to apply additional protections to your data such as encryption or fine-grained access controls
- ☐ Backup your Azure Storage data
- ☐ Evaluate whether to use token management to prevent the use of personal access tokens
- ☐ Use workspace bindings to isolate sensitive datasets and environments
- ☐ Restrict cluster creation rights and use compute policies to enforce data access patterns and control costs
- ☐ Review and configure workspace admin settings
- ☐ Consider whether to apply restrictions on the use of libraries, models and code
- ☐ Consider the use of Enhanced Security Monitoring or the Compliance Security Profile
- ☐ Provision infrastructure via infrastructure-as-code
- ☐ Monitor system activities via Azure logs
- ☐ Design, implement & test a Disaster Recovery strategy if you have strong business continuity requirements

4. Azure Databricks threat models

Customers who are particularly security conscious may want to understand the threat models that might apply to platforms like Azure Databricks and the controls they can leverage to mitigate specific risks. If you are looking to ensure that you're following best practices and don't have specific security concerns you are looking to protect against, you can skip this section and focus on the checklists provided above. The most common threat categories that come up in customer conversations are:

1. Account takeover or compromise
2. Data exfiltration
3. Insider threats
4. Supply chain attacks
5. Potential compromise of Databricks
6. Ransomware attacks
7. Resource abuse such as crypto mining

This section addresses common questions about these risks, discusses probabilities, and provides mitigation strategies.

Account takeover or compromise

Risk description
Azure Databricks is a general-purpose compute platform that customers can set up to access critical data sources. If credentials belonging to a user at one of our customers were compromised by phishing, brute force, or other methods, an attacker might get access to all of the data accessible from the environment.

Probability
Without proper protections, account takeover can be an effective strategy for an attacker. Fortunately, it is easy to apply strategies that dramatically reduce the risk.

Protect	Detect	Respond
• 1.1 Leverage multi-factor authentication for all user access • 1.2 Use SCIM to synchronize users and groups and correctly deprovision users when they leave your organization • 1.7 Use OAuth or Entra ID token authentication to ensure that short-lived tokens are used for access • 1.13 Store and use secrets securely to protect user and system credentials • 1.8 Enforce token management to disable personal access tokens or set a maximum lifetime for them • 3.2 Configure IP access lists for your account, workspaces and Delta shares to restrict access to trusted public networks • 3.4 Use Azure PrivateLink to restrict access to trusted private networks • 3.5 Implement network exfiltration protections to protect against data exfiltration following a successful account takeover attack • 3.8 Restrict access to valuable codebases to only trusted networks	• 5.1 Leverage system tables to identify failed authentication, authorization and access attempts. Please refer to this blog for some examples • 5.10 Implement DevSecOps processes to identify credentials in your code	• 1.2 Use SCIM to manage users and groups to disable / remove potentially compromised users • 1.7 Use OAuth or Entra ID token authentication to delete OAuth secrets, deactivate and remove service principals • 1.8 Enforce token management to revoke tokens and/or disable token authentication • 5.3 Enable verbose audit logging so that the actions of potentially compromised accounts can be investigated

Data exfiltration

Risk description
If a malicious user or an attacker is able to log into a customer's environment, they may be able to exfiltrate sensitive data and then store it, sell it, or ransom it.

Probability
While the probability of this type of attack is generally low because it presumes either a malicious insider or compromised account, it is not uncommon for these types of attackers to attempt to exfiltrate and then leverage data.

Protect	Detect	Respond
• 1.11 Use service principals to run administrative tasks and production workloads so that wherever possible users do not need direct access to sensitive data • 2.3 Plan your data isolation model so that sensitive data is protected by the appropriate level of isolation • 2.3 Avoid storing production data in DBFS • 2.5 Configure Azure Storage firewalls to restrict access to trusted networks • 2.6 Prevent anonymous read access & apply other protections • 2.12 Configure a Delta Sharing recipient token lifetime • 2.13 Additionally encrypt sensitive data at rest using Advanced Encryption Standard (AES) • 2.14 Leverage data exfiltration prevention settings within the workspace • 2.15 Use Clean Rooms to collaborate in a privacy-safe environment • 3.3 Configure IP access lists to protect your Delta Shares • 3.5 Implement network exfiltration protections to restrict outbound access to trusted destinations • 3.6 Isolate sensitive workloads into different networks • 3.7 Configure a firewall for serverless compute access • 4.2 Isolate sensitive workloads into different workspaces • 4.3 Assign Unity Catalog securables to specific workspaces to restrict access to securables that may contain sensitive data • 5.5 Restrict usage to trusted code repositories so that code cannot be easily exfiltrated from the environment	• 5.1 Leverage system tables to identify repeated failed authorisation requests, high numbers of reads and writes and changes to account and workspace settings that protect against exfiltration. Please refer to this blog for some examples • 5.2 Monitor system activities via Azure logs to identify failed & suspicious Entra ID, data access and network access attempts	• 1.2 Use SCIM to manage users and groups to disable / remove accounts that are under investigation • 5.3 Enable verbose audit logging so that the actions relating to potential data exfiltration attempts can be investigated

Insider threats

Risk description
High-performing engineers and data professionals will generally find the best or fastest way to complete their tasks, but sometimes that may do so in ways that create security impacts to their organizations. One user may think their job would be much easier if they didn't have to deal with security controls, or another might copy some data to a public storage account or other cloud resource to simplify sharing of data. We can provide education for these users, but companies should also consider providing guardrails.

Probability
Given the large number of ways that security protocols can be avoided, there is significant variability in the likelihood and impact of risks in this category. That said, most security professionals identify this as a significant potential risk to organizations.

Protect	Detect	Respond
• 1.2 Use SCIM to synchronize users and groups, helping to ensure that users have the correct level of access • 1.3 Limit the number of admin users • 1.4 Enforce segregation of duties between administrative accounts • 1.5 Restrict workspace admins • 1.12 Use compute that supports user isolation so that users & workloads are isolated, even on shared compute • 1.13 Store and use secrets securely to protect user and system credentials • 2.7 Enable soft deletes and other data protection features so that incorrectly overwritten or deleted data can be recovered • 2.8 Backup your Azure Storage data so that full datasets can be recovered when necessary • 2.13 Additionally encrypt sensitive data at rest using Advanced Encryption Standard (AES) • 3.5 Implement network exfiltration protections as the safeguards they provide against accidental insider exposure are similar to those provided against a malicious attacker • 3.8 Restrict access to valuable codebases to only trusted networks • 5.4 Manage code versions with Git folders so that code is backed up outside of the platform • 5.5 Restrict usage to trusted code repositories • 5.6 Provision infrastructure via infrastructure-as-code so that	• 4.8 Use Enhanced Security Monitoring or Compliance Security Profile to identify and alert on suspicious activity that might indicate an attempt to break out of the environment. Please refer to this blog for some examples • 5.1 Leverage system tables to identify destructive activities (high number of deletes within a session) and privilege escalation attempts (high number of permission changes within a session). Please refer to this blog for some examples • 5.2 Monitor system activities via Azure logs to identify failed & suspicious data access and network access attempts	• 1.2 Use SCIM to manage users and groups and disable / remove the accounts of potential insider threats • 4.10 Implement and test a Disaster Recovery strategy to recover your data if needed • 5.3 Enable verbose audit logging so that the actions of potential accidental or malicious insiders can be investigated

Protect	Detect	Respond
environments can be recreated if necessary, and manual changes to production environments are not allowed • 5.7 Manage code via CI/CD so that only approved code can be run in production environments		

Supply chain attacks

Risk description
Historically, supply chain attacks have relied upon injecting malicious code into software libraries. That code is then executed without the knowledge of the unsuspecting target. More recently, however, we have started to see the emergence of AI model and data supply chain attacks, whereby the model, its weights or the data itself is maliciously altered.

Probability
Without proper protections, supply chain attacks could be an effective strategy for an attacker. Fortunately, it is easy to apply protection strategies that dramatically reduce this risk.

Protect	Detect	Respond

• 3.5 Implement network exfiltration protections as the safeguards they provide against supply chain attacks are similar to those provided against a malicious attacker
• 4.2 Isolate sensitive workloads into different workspaces so that users have more freedom to experiment with libraries in sandbox environments, but only trusted libraries are used in production
• 5.5 Restrict usage to trusted code repositories so that untrusted code cannot be easily brought into the environment
• 5.7 Manage code via CI/CD so that only scanned and approved code can be run in production environments
• 5.8 Control library installation so that only scanned and approved libraries can be used for sensitive workloads
• 5.9 Use models and data from only trusted or reputable sources
• 5.10 Implement DevSecOps processes to automatically scan code, libraries, dependencies, models and model weights
• 5.11 Use lakehouse monitoring to identify changes to the quality and consistency of important datasets which may indicate data supply chain attacks such as data poisoning and label flipping
	• 4.8 Use Enhanced Security Monitoring or Compliance Security Profile to identify and alert on suspicious activity that might indicate an attempt to break out of the environment. Please refer to this blog for some examples
	• 5.10 Implement DevSecOps processes to automatically scan code, libraries, dependencies, models and model weights
		• 5.1 Leverage system tables and search to identify the use of libraries with known vulnerabilities. Please refer to the following blogs for some examples:
		  ○ Scanning for Arbitrary Code in a Databricks Workspace
		  ○ Monitoring Notebook Command Logs With Static Analysis Tools
		• 5.3 Enable verbose audit logging to identify library installs via notebook commands
		• 5.8 Control library installation to disallow access to libraries with known vulnerabilities

Potential compromise of the cloud provider and/or Azure Databricks

Risk description
Security-minded customers sometimes voice a concern that Azure Databricks itself might be compromised, which could result in the compromise of their environment.

Azure Databricks is a Microsoft product, and the control and serverless compute planes are running in a Microsoft-managed subscription, so customers should also consider the Microsoft security program whilst evaluating this risk. Please see the Microsoft Security and Trust Center for more information and reach out to your Azure representative as required.

Probability
Azure Databricks invests considerable resources into securing its Data Intelligence Platform and has a robust security program designed to minimize the risk of such an incident – see our Security and Trust Center for an overview of the program and relevant security controls. However, the risk for any company is never completely eliminated.

Protect	Detect	Respond

• Review the Microsoft and Databricks Security & Trust Centers and consider any necessary process controls
• 4.9 Control & monitor workspace access for Azure Databricks personnel
	• 5.1 Leverage system tables to monitor the activities of Azure Databricks employees that you grant access to your environment. Please refer to this blog for some examples
	• 5.2 Monitor system activities via Azure logs to identify
	  ○ Abnormal provisioning activity
	  ○ Suspicious or failed Entra ID access attempts
	  ○ Suspicious or failed data access attempts
		• Review the Microsoft and Databricks Security & Trust Centers and consider any necessary process controls
		• 5.1 Leverage system tables to monitor the activities of Azure Databricks employees that you grant access to your environment. Please refer to this blog for some examples
		• 5.3 Enable verbose audit logging to monitor the activities of Azure Databricks employees that you grant access your environment.
		• Prepare "worst case scenario" controls in the event of an active compromise:
		  ○ Remove access to your data by revoking your customer-managed keys for managed services (not guaranteed to be a reversible operation) and storage
		  ○ Remove access to your data by revoking managed identity permissions to your storage

Ransomware attacks

Risk description
Ransomware is a type of malware designed to deny an individual or organization access to their data, usually for the purposes of extortion. Encryption is often used as the vehicle for this attack. In recent years, there have been several high profile ransomware attacks that have brought large organizations to their knees.

Probability
The vast majority of data is stored within customers' own storage accounts, which would present a far more appealing target for ransomware attacks. Therefore, while we provide a brief summary here, the most important security controls are those that customers configure for their own storage.

Protect	Detect	Respond
• 2.1 Centralise data governance with Unity Catalog to ensure that only time-bound, down-scoped tokens are used to access data • 2.5 Configure Azure Storage firewalls to protect your resources from untrusted networks • 2.7 Enable soft deletes and other data protection features so that incorrectly overwritten, deleted or corrupted data can be recovered • 2.8 Backup your Azure Storage data so that full datasets can be recovered when necessary • 2.10 Configure customer-managed keys for storage so that you have more control and visibility over the encryption keys used to protect your data • 2.11 Use Delta Sharing to ensure that only read-only, time-bound, down-scoped tokens are used to access data • 3.7 Configure a firewall for serverless compute access to protect your resources from untrusted networks • 3.8 Restrict access to valuable codebases to only trusted networks • 5.6 Provision infrastructure via infrastructure-as-code so that manual changes to production environments are not allowed	• 5.2 Monitor system activities via Azure logs to identify suspicious or failed Entra ID, data or CMK access attempts and attempts to modify storage account configurations • 5.10 Implement DevSecOps processes to identify credentials in your code	• 2.7 Enable soft deletes and other data protection features to restore incorrectly overwritten, deleted or corrupted data • 2.8 Backup your Azure Storage data and restore full datasets where necessary • 2.10 Configure customer-managed keys for storage and put a process in place to rotate and revoke keys where necessary • 4.10 Implement and test a Disaster Recovery strategy to recover your data if required

Resource abuse

Risk description
Azure Databricks can deploy large amounts of compute power. As such, it could be a valuable target for crypto mining if a customer's user account were compromised.

Probability
This has not been a prominent activity in practice, but customers will sometimes bring up this concern.

Protect	Detect	Respond
• 1.9 Restrict cluster creation rights • 1.10 Use compute policies to restrict the maximum size and types of compute • 5.8 Control library installation to reduce the risk of supply chain attacks that are designed to result in resource abuse • 5.15 Use Azure Policy to limit the resources that can be deployed	• 5.1 Leverage system tables to monitor billable usage • 5.2 Monitor system activities via Azure logs to identify abnormal provisioning activity • 5.10 Implement DevSecOps processes to identify credentials in your code • 5.13 Use tagging as part of your cost monitoring and charge-back strategy • 5.14 Use budgets to monitor account spending	• 1.2 Use SCIM to manage users and groups to disable / remove accounts that are under investigation • 5.3 Enable verbose audit logging so that the actions relating to resource abuse attempts can be investigated

Appendices

Appendix A - Security configuration reference

The security configurations referenced throughout this document are described in more detail below. For ease of reference, these security configurations have been grouped into the following overarching security, compliance, and privacy principles:

- Manage identity and access using least privilege
- Protect data in transit and at rest
- Secure your network and protect endpoints
- Meet compliance and data privacy requirements
- Monitor system security

Manage identity and access using least privilege

The practice of identity and access management (IAM) helps you ensure that the right people can access the right resources. IAM addresses the following aspects of authentication and authorization: account management including provisioning, identity governance, authentication, access control (authorization), and identity federation.

1.1 Leverage multi-factor authentication

Azure Databricks supports Microsoft Entra ID conditional access, which allows administrators to control the conditions under which users are permitted to sign in to their Azure resources. Conditional access policies can restrict sign-in to trusted networks and can require multi-factor authentication (MFA).

For the highest security environments, Azure Databricks also advocates where possible for the use of physical authentication tokens such as FIDO2 keys. These keys augment traditional Multi-Factor authentication by requiring interaction with a physical token that cannot be compromised.

It's important to note that Microsoft Entra ID conditional access applies at the point of authentication with Microsoft Entra ID. It is not enforced for users who have already authenticated with Entra ID and subsequently change networks, or who are using alternative methods of authentication such as Personal Access Tokens. Therefore, for comprehensive network access controls Azure Databricks recommends that customers combine Microsoft Entra ID conditional access with the use of IP access lists and/or Azure Private Link.

1.2 Use SCIM to synchronize users and groups

SCIM (System for Cross-domain Identity Management) allows you to sync users and groups between Microsoft Entra ID and Azure Databricks. There are three major benefits of this approach:

1. When you remove a user, the user is automatically removed from Azure Databricks.
2. Users can also be disabled temporarily via SCIM. Customers have used this capability for scenarios where customers believe that an account may be compromised and need to investigate
3. Groups are automatically synchronized

Azure Databricks recommends that customers sync users and groups from Microsoft Entra ID to your Azure Databricks account, enable identity federation and use SCIM provisioning to manage all users and groups within your account.

1.3 Limit the number of admin users

As in most systems, administrators within Databricks have elevated privileges that should only be extended to a trusted few within an organization. Where possible, use automation via Service Principals to perform administrative tasks, preferably via infrastructure-as-code. This recommendation applies to all Azure Databricks admin roles.

It's also important to note that as part of the Azure RBAC model, users or service principals that are given Contributor or above permissions to the Resource Group for an Azure Databricks workspace automatically become administrators when they login. Therefore, the same considerations outlined above should be applied to Azure portal users, service principals and managed identities too.

1.4 Enforce segregation of duties between administrative accounts

It is a general best practice across all of security that an administrator should not use their privileged accounts to perform day-to-day tasks. Azure Databricks recommends that customers should maintain a segregation of duties between user accounts, ensuring that:

- The same user does share multiple highly privileged roles (such as account and metastore admin)
- Databricks administrators who are also normal users of the Azure Databricks platform use a separate user account for administrative versus day-to-day tasks

Where possible, use automation via Service Principals to perform all administrative tasks, preferably via infrastructure-as-code. This recommendation applies to all Azure Databricks admin roles.

As above, it's important to note that as part of the Azure RBAC model, users or service principals that are given Contributor or above permissions to the Resource Group for an Azure Databricks workspace automatically become administrators when they login.

1.5 Restrict workspace admins

By default, workspace admins can change the job owner or run as setting and generate on-behalf-of tokens for any service principal in their workspace. Azure Databricks recommends configuring the restrict workspace admins setting to prevent this.

1.6 Manage access according to the principle of least privilege

Within Azure Databricks there are different access control systems for different securable objects. Azure Databricks recommends assigning ACLs according to the principle of least privilege, and assigning them to groups rather than directly to users. For Unity Catalog securables, manage access at the lowest level in the inheritance model. This proposal for persona based access control should help you to get started.

1.7 Use OAuth or Azure Entra ID token authentication

Where possible customers should only use OAuth user-to-machine (U2M), machine-to-machine (M2M) or Azure Entra ID authentication. OAuth reduces risk because U2M requires users to authenticate as they would via the UI and for M2M the credential in memory will typically be a short-lived access token. Whilst most code will need a way to read the secret in order to request a new access token, the secret can be stored securely (for example in a service like Azure Key Vault) and pulled down only when a new access token is requested. Azure Entra ID reduces risk for similar reasons.

1.8 Enforce token management

Customers can use the Token Management API or UI controls to enable or disable personal access tokens (PATs) for REST API authentication, limit the users who are allowed to use PATs, set the maximum lifetime for new tokens, and manage existing tokens. Where possible we would encourage highly secure customers to use Azure Entra ID or OAuth token authentication. Where this is not possible, we would recommend that they provision a short maximum token lifetime for new tokens within a workspace. Customers can use the notebook provided here to assess personal access token usage within their Azure Databricks account.

1.9 Restrict cluster creation rights

Using either compute policies or the cluster creation entitlement, admins can define which users or groups within the organization are able to create clusters.

Compute permissions allow you to specify which users can perform which actions on a given cluster. Note that using the correct cluster isolation level is a consideration here too, and shared access mode clusters, SQL warehouses and serverless compute should be preferred where possible.

1.10 Use compute policies

Azure Databricks admins can control many aspects of the clusters that are spun up, including size of clusters, available instance types, runtime versions and Spark configuration settings using compute policies. Admins can configure multiple compute policies, allowing certain groups of users to create small clusters, some groups of users to create large clusters, and other groups to only use existing clusters.

1.11 Use service principals to run administrative tasks and production workloads

It is against security best practices to tie production workloads to individual user accounts, and so we recommend configuring Service Principals within Azure Databricks. Service Principals separate administrator and user actions from the workload and prevent workloads from being impacted if a user leaves an organization. You can configure jobs as well as automation tools to run as a service principal.

Within Azure Databricks, Service principals can either be Azure Databricks managed service principals or Microsoft Entra ID managed service principals. Azure Databricks recommends that you use Azure Databricks managed service principals for Azure Databricks automation and that you use Microsoft Entra ID managed service principals in cases where you must authenticate with Azure Databricks and other Azure resources at the same time.

1.12 Use compute that supports user isolation

Customers should use shared or assigned access mode clusters, SQL warehouses or serverless compute at all times, with a preference towards shared access mode, SQL warehouses and serverless. These compute types apply isolation boundaries between users & workloads.

If No isolation shared clusters must be used, then customers should enable admin protection so that admin credentials are protected in an environment that is shared with other users.

1.13 Store and use secrets securely

Integrating with heterogeneous systems requires managing a potentially large set of credentials and safely distributing them across an organization. Instead of directly entering your credentials into a notebook, use Azure Databricks secrets to store your credentials and reference them in notebooks and jobs. Azure Databricks secret management allows users to use and share credentials within Databricks securely.

Customers can choose whether to store their secrets in Azure Databricks or Azure Key Vault, and then configure access control lists to define which users and groups can access them.

It's important to note that even if customers use Azure Key Vault to store their secrets, access controls still need to be defined within Azure Databricks. This is because the same service identity is used to retrieve the secret for all users of an Azure Databricks workspace

Protect data in transit and at rest

Classify your data into sensitivity and criticality levels and use mechanisms such as encryption, tokenization, and access control where appropriate.

2.1 Centralise data governance with Unity Catalog

Unity Catalog offers a unified governance layer for data and AI within the Azure Databricks Data Intelligence Platform. With Unity Catalog, organizations can seamlessly govern their structured and unstructured data, machine learning models, notebooks, dashboards and files on any cloud or platform. This unified approach to governance accelerates data and AI initiatives while simplifying regulatory compliance.

2.2 Use Azure Managed Identities to access storage

Azure Databricks recommends using Azure managed identities in Unity Catalog to access data stored in your Azure storage accounts. Once you have granted this access, Databricks recommends configuring storage firewalls to prevent access from untrusted networks (note that the compute plane is a trusted network and should be granted access via a private or service endpoint). See (Recommended) Configure trusted access to Azure Storage based on your managed identity and Configure a firewall for serverless compute access for more details.

2.3 Plan your data isolation model

Unity Catalog gives you the ability to choose between centralized and distributed governance models, as well as apply varying levels of isolation between datasets. Databricks recommends that you plan your data isolation model upfront, following the best practice recommendations provided.

2.4 Avoid storing production data in DBFS

By default, DBFS is a filesystem that is accessible to all users of the given workspace and can be accessed via API. This is not necessarily a major data exfiltration concern as you can limit access to accessing data via the DBFS API or the Azure Databricks CLI using IP access lists or private network access. However, as use of Azure Databricks grows and more users join a workspace, those users would have access to any data stored in DBFS, creating the potential for undesired information sharing. Azure Databricks recommends that customers do not store production data in DBFS.

2.5 Configure Azure Storage firewalls

There are two main types of Azure storage accounts within an Azure Databricks deployment: the managed storage account that gets created automatically when you deploy an Azure Databricks workspace and any additional storage accounts in which you store your data.

For all storage accounts that store sensitive data, Azure Databricks recommends restricting access to trusted networks (such as your virtual network and serverless compute) and managed identities with a storage firewall.

The managed storage account associated with an Azure Databricks workspace is protected by a Deny Assignment which prohibits any direct external access to the storage; it can only be accessed via the Azure Databricks workspace. However, for highly secure deployments, Azure Databricks recommends enabling firewall support for the workspace storage account too.

> If you are using external links to retrieve large data sets via the SQL Statement Execution API, Azure Databricks recommends that you configure network restrictions on your storage accounts. See Security best practices for more information.

2.6 Prevent anonymous read access & apply other protections

Customers should review the storage accounts that they manage against the Azure security baseline for Storage. In particular they should ensure that Anonymous read access is not allowed, but they may also consider other protections, such as the use of a customer-managed key.

2.7 Enable soft deletes and other data protection features

Azure storage provides a number of features that allow you to backup and recover your data if needed. Customers should consider the various options available and apply as necessary to meet their requirements:

- Resource locks
- Soft deletes for containers
- Soft deletes for blobs
- Data redundancy

Please refer to the Best practices for Azure Storage data protection, backup, and recovery for an exhaustive list.

2.8 Backup your Azure Storage data

Create regular backups of your data, allowing you to recover it from accidental deletion or corruption. Note that Azure Backup, blob versioning and point-in-time restore for blobs are not currently supported for ADLS Gen2. Microsoft recommends copying data to a second account via Azure Storage object replication or tools like AzCopy. Delta cloning can also be used to create backups of your data.

2.9 Configure customer-managed keys for managed services

Configure a customer-managed key (CMK) for scoped data stored within the Azure Databricks control plane and serverless compute plane, such as:

- Notebooks
- SQL queries
- SQL query history
- Secrets
- Personal access tokens (PAT) or other credentials
- Vector search indexes and metadata

Azure Databricks requires access to this key for ongoing operations. You can revoke access to the key to prevent Azure Databricks from accessing encrypted data within the control plane (or in our backups). This is like a “nuclear option” where the workspace ceases to function, but it provides an emergency control for extreme situations.

For more information on using a customer-managed key (CMK) with Databricks please refer to Customer-managed keys for encryption.

2.10 Configure customer-managed keys for storage

Configure a customer-managed key for scoped data stored within the compute and data planes, such as:

- The Azure managed disks attached in the customer-managed compute plane
- The Azure storage account associated with a Databricks workspace
- The Azure storage accounts managed or accessed by Unity Catalog

Azure Databricks requires access to this key for ongoing operations, but a customer-managed key helps meet compliance requirements and allows you to revoke access if required.

For more information on using a customer-managed key (CMK) with Azure Databricks please refer to Customer-managed keys for encryption.

> Serverless compute resources do not use customer-managed keys for managed disk encryption on compute nodes. Managed disks for serverless compute resources are short-lived and tied to the lifecycle of the serverless workload. When compute resources are stopped or scaled down, the VMs and their storage are destroyed.

2.11 Use Delta Sharing

Delta Sharing is the first open source approach to data sharing across data, analytics and AI. Customers can share live data across platforms, clouds and regions with strong security and governance. Follow the Security Best Practices for Delta Sharing when sharing sensitive data.

2.12 Configure a Delta Sharing recipient token lifetime

When enabling Delta Sharing for a metastore, always ensure that recipient tokens are set to expire within a timescale (seconds, minutes, hours or days) that is proportional to the sensitivity of the data that might be shared.

2.13 Additionally encrypt sensitive data at rest using Advanced Encryption Standard (AES)

Azure Databricks supports Advanced Encryption Standard (AES) encryption to additionally encrypt columns of sensitive data at rest. Customers can use the aes_encrypt and aes_decrypt functions to convert between plaintext and ciphertext, using secrets to securely store the cryptographic keys. Additionally encrypting sensitive data at rest adds another layer of protection in the event that the underlying storage account and its encryption keys or cryptography become compromised.

2.14 Leverage data exfiltration prevention settings within the workspace

Azure Databricks workspace admins can leverage a variety of settings that provide protection. Most admin controls are simple enable/disable buttons. Some of the most important ones are:

- ☐ Notebook results download
- ☐ Notebook exporting
- ☐ SQL results download
- ☐ MLflow run artifact download
- ☐ Results table clipboard features
- ☐ FileStore Endpoint

2.15 Use Clean Rooms to collaborate in a privacy-safe environment

Azure Databricks Clean Rooms allow you to easily collaborate with your customers and partners in a secure environment in a privacy-safe way. Clean Rooms can enable collaboration whilst protecting against unauthorized access or inadvertent data leakage.

For more information please refer to What is Azure Databricks Clean Rooms?

Secure your network and protect endpoints

Secure your network and monitor and protect the network integrity of internal and external endpoints through security appliances or cloud services like firewalls.

3.1 Use Secure Cluster Connectivity (No Public IP)

With secure cluster connectivity enabled, customer virtual networks require no open inbound ports from external networks and Databricks cluster nodes have no public IP addresses. Azure Databricks recommends this configuration for all Azure Databricks workspaces because it significantly reduces the attack surface and hardens the security posture.

3.2 Deploy Azure Databricks into your own Azure virtual network

For non-serverless workloads, Azure Databricks requires the use of a virtual network within the customer's Azure subscription. Azure Databricks recommends that customers deploy into their own virtual network so they can integrate Azure Databricks into their existing network architecture, including routing traffic through their own network enforcement points (such as a firewall) and securely accessing data via Azure Private Link.

> For serverless workloads, the compute plane network is managed and secured by Azure Databricks. One less security configuration for you to manage!

3.3 Configure IP access lists

IP access lists restrict the IP addresses that can be used to access Azure Databricks by checking if the user or API client is coming from a trusted IP address range such as a VPN or office network. Established user sessions do not work if the user moves to a bad IP address, such as when disconnecting from the VPN. Azure Databricks recommends that customers configure IP access lists for their Azure Databricks account, workspaces and Delta Sharing recipients.

3.4 Use Azure PrivateLink

Azure Private Link allows customers to set up end-to-end private networking for their Azure Databricks Data Intelligence Platform. Private Link can be configured between Azure Databricks users and the control plane, between the control plane and the compute plane, and between the compute plane and Azure services.

Configuring Azure Private Link for back-end and front-end connections ensures that your Azure Databricks workspaces can only be accessed over that dedicated and private channel.

For non-serverless workloads, customers can use private endpoints to connect from their virtual networks to their storage accounts.

For serverless workloads, customers can create network connectivity configurations that use dedicated private endpoints to connect to their storage accounts.

For more information on using Azure Private Link with Azure Databricks please refer to Enable Azure Private Link back-end and front-end connections and Configure private connectivity from serverless compute.

> For serverless workloads, networking between the control and compute planes is managed by Azure Databricks using either Azure PrivateLink or the Azure network secured with mutual TLS authentication and firewall policies that limit access to only valid IPs. One less security control for you to worry about!

3.5 Implement network exfiltration protections

By default, compute plane hosts within your Azure environment have unrestricted outbound network access to specific services & ports. If you deploy into your own virtual network, you can restrict outbound traffic using a firewall. Azure Databricks has published a blog post that describes how to do this using Azure Firewall, but it can be generalized to other network security tools using details provided in the Azure Databricks documentation.

The TLS connections between the control plane and the compute plane cannot be broken, and so it's not possible to use a technology like SSL or TLS inspection. The custom TLS certificate that would be needed cannot be pre-loaded on the Azure Databricks VHD that is built for all customers.

3.6 Isolate Azure Databricks workspaces into different networks

Customers can deploy multiple workspaces into the same virtual network (VNET), but for sensitive workloads this is not recommended. Customers should isolate these workloads into their own workspace deployed into their own virtual network.

If shared networking resources like DNS are required, Azure Databricks strongly recommends you follow the Azure best practices for hub and spoke model. Use VNet peering to extend the private IP space of the workspace VNet to the hub while keeping spokes isolated from each other.

If you have other resources in the VNet or use peering, Databricks strongly recommends that you add Deny rules to the network security groups (NSGs) that are attached to other networks and subnets that are in the same VNet or are peered to that VNet. Add Deny rules for connections for **both inbound and outbound connections** so they limit connections both *to* and *from* Azure Databricks compute resources. If your cluster needs access to resources on the network, add rules to allow only the minimal amount of access required to meet the requirements. For more information see Shared resources and peering and Network security group rules.

For serverless compute, customers can use network connectivity configurations (NCCs) to manage logically related networks. Customers should create NCCs based on their desired logical separation of serverless date planes, while bearing the documented limits in mind.

3.7 Configure a firewall for serverless compute access

For serverless workloads, customers can create network connectivity configurations that use a specific set of stable subnet ids or dedicated Private Link endpoints to connect to their resources. Customers can then protect these resources by allowlisting only these connections.

3.8 Restrict access to valuable codebases to only trusted networks

Azure Databricks recommends that customers restrict access to valuable codebases to only trusted networks. In order to use these code repositories within Azure Databricks, customers can apply either public or private networking controls.

3.9 Use Virtual network encryption

Azure Virtual Network encryption allows you to seamlessly encrypt and decrypt traffic between Azure Virtual Machines via a DTLS tunnel. This can be a great way to ensure that traffic is always encrypted within internal networks, whilst incurring minimal performance impact.

Meet compliance and data privacy requirements

You might have internal (or external) requirements that require you to control the data storage locations and processing. These requirements vary based on systems design objectives, industry regulatory concerns, national law, tax implications, and culture. Be mindful that you might need to obfuscate or redact personally identifiable information (PII) to meet your regulatory requirements. Where possible, automate your compliance efforts.

4.1 Restart compute on a regular schedule

Azure Databricks compute clusters are ephemeral. Upon launch they will automatically use the latest available base image and container image. Users cannot choose an older version that may have security vulnerabilities, with the exception of out-of-support container images which are hidden from the UI but can be manually configured or may have been configured on a cluster before the release was hidden.

Customers are responsible for making sure that clusters are restarted periodically. Azure Databricks does not live-patch systems--when a cluster is restarted and newer system images or containers are available, the system will automatically use the latest available images and containers.

> Automatic cluster restart is automatically enabled where the compliance security profile is enabled. One less security control for you to manage!
>
> Serverless compute is limited to a maximum of 7 days of total uptime before being recycled seamlessly in the background. One less security control for you to think about!

4.2 Isolate sensitive workloads into different workspaces

While Azure Databricks has numerous capabilities for isolating different workloads within a workspace, such as access control lists and Unity Catalog privileges and securable objects, the strongest isolation control is to move sensitive workloads to a different workspace. This sometimes happens when a customer has very different teams (for example, a security team and a marketing team) who must both analyze very different data.

4.3 Assign Unity Catalog securables to specific workspaces

If you use workspaces to isolate users and data, you may want to limit access to Unity Catalog securables to specific workspaces in your account. These assignments (also known as bindings) can be used to restrict access to catalogs, storage credentials and external locations that may access or contain sensitive data to specific workspaces.

Bindings can also be used to provide read-only access, which can be useful in certain scenarios (for example by giving a data scientist read-only access to production datasets for Exploratory Data Analysis).

4.4 Implement fine-grained access controls

For sensitive datasets, implement fine-grained access controls via row filters and column masks.

4.5 Apply tags

Apply tags to sensitive datasets so that they can be easily discovered, identified and handled appropriately. Tags can be automatically applied via Lakehouse Monitoring, and can be used to support fine-grained access controls including Attribute-based access controls (ABAC) which is in preview.

4.6 Use lineage

Use lineage within Unity Catalog to track the movement of sensitive data, improving data governance and allowing you to more accurately meet regulatory data subject requests.

4.8 Use Enhanced Security Monitoring or Compliance Security Profile

Enhanced Security Monitoring (ESM) and Compliance Security Profile (CSP) provides the most secure baseline for Azure Databricks deployments.

Enhanced Security Monitoring provides:

1. A VM with enhanced CIS Level 1 hardening
2. Behavior-based malware monitoring and file integrity monitoring (Capsule8)
3. Malware and anti-virus detection (ClamAV)
4. Qualys vulnerability reports from a representative host OS

The Compliance Security Profile includes all the benefits above, and layers on additional security controls required to meet compliance requirements:

1. Automatic cluster updates
2. HIPAA and PCI-DSS compliant features and controls

4.9 Control & monitor workspace access for Azure Databricks personnel

Azure Databricks personnel cannot access customer workspaces or the production multi-tenant environments without customer consent. If you raise a support request, you can grant Azure Databricks personnel temporary access to your workspaces in order to investigate an outage or security event, or to support your deployment.

Azure Databricks recommends that customers configure workspace access for Azure Databricks personnel to be Not enabled by default, and only grant access as needed on a time-bound basis. Azure Databricks also recommends that customers monitor such access via their system tables.

4.10 Implement and test a Disaster Recovery strategy

While Azure Databricks doesn't offer disaster recovery (DR) services, customers can implement their own DR procedures for their data stored in Azure, using either cloud native backup services or Delta cloning. Customers can also implement cross-region resiliency for mission critical workloads via Delta Live Tables.

Where customers need to be able to failover *entirely* to a separate DR site, they can use Azure Databricks capabilities to create a cold (on standby) workspace in another region. Please refer to our disaster recovery guide for more information.

4.11 Consider the use of Azure Confidential Compute

For highly sensitive workloads, such as those that de-identify personal data, you may want to consider the use of Azure Confidential Compute to protect the data in-use.

Please refer to Azure confidential computing VMs for more information.

Monitor system security

Use automated tools to monitor your application and infrastructure. To scan your infrastructure for vulnerabilities and detect security incidents, use automated scanning in your continuous integration and continuous deployment (CI/CD) pipelines.

5.1 Leverage system tables

System tables serve as a centralized operational data store, backed by Delta Lake and governed by Unity Catalog. System tables can be used for a variety of different purposes, from cost monitoring to audit logging. Azure Databricks recommends that customers configure system tables and set up automated monitoring and alerting to meet their needs. The blog post Improve Lakehouse Security Monitoring using System Tables in Databricks Unity Catalog is a good starting point.

Customers that are using Enhanced Security Monitoring or the Compliance Security Profile can monitor and alert on suspicious activity detected by the behavior-based malware and file integrity monitoring agents.

5.2 Monitor system activities via Azure logs

It is a security adage that you cannot trust the system to tell you when it is compromised, you must be able to observe the system from the outside. System tables audit logs are an extremely valuable feature for monitoring what users do, but many customers want an outside resource to help monitor that Azure Databricks itself doesn't do something wrong.

Azure logs such as resource logs, activity logs, Entra ID and VNET / NSG flow logs provide a great mechanism for observing the actions of Azure Databricks (and users) in the compute and data planes. They provide visibility into:

- Resource creation, to help identify bitcoin mining and also as a control for billing
- Outbound network connections, to help identify data exfiltration*
- Subscription level events such as API calls, to help identify account compromise.
- Access to data using Unity Catalog as a secure data broker

Most customers have favorite tools in place to analyze cloud provider log data, but you can also analyze this in Azure Databricks.

Please see the Azure documentation for more information.

> *If you have deployed a network level protection such as a firewall, then monitoring your firewall traffic logs is likely to be the best way to achieve this.

5.3 Enable verbose audit logging

In some highly regulated domains it is a requirement to track every command that a user has run against the system. On Azure Databricks this can be achieved via verbose audit logging. Once configured, audit logs will be recorded in system tables whenever a query or command is run within your workspace.

5.4 Manage code versions with Git folders

Azure Databricks recommends that customers use Git folders to manage and protect their source code, as per widely accepted software development best practices.

5.5 Restrict usage to trusted code repositories

A workspace admin can restrict which remote repositories users can clone from and commit & push to. This helps prevent exfiltration of your code and infiltration of untrusted code.

5.6 Provision infrastructure via infrastructure-as-code

Using infrastructure-as-code (IaC) to provision infrastructure provides a number of benefits, including but not limited to:

- Reduced likelihood of configuration errors due to human error
- Reduced likelihood of configuration drift where secure baseline templates are developed
- Automatic reversal of configuration drift the next time the IaC tool runs
- Reduced likelihood of outages due to infrastructure being accidentally modified or deleted
- Faster recovery times in the event of an environment needing to be recreated from scratch, such as in a disaster recovery / business continuity scenario
- Reduced number of administrative users
- Reduced number of administrative users who also have day-to-day permissions

Azure Databricks recommends that customers use infrastructure-as-code to provision both their cloud and Azure Databricks infrastructure, preferably via service principals whose credentials are only made available when needed to highly trusted individuals.

> Our Security Reference Architecture (SRA) Terraform templates make it easy to deploy Databricks workspaces that follow these Security Best Practices!

5.7 Manage code via CI/CD

Mature organizations build and deploy production workloads using CI/CD, allowing them to better manage user permissions to production environments, integrate code scanning, perform linting, and more. When there is highly sensitive data analyzed, a CI/CD process can also allow scanning for known scenarios such as hard coded secrets.

5.8 Control library installation

By default, Azure Databricks allows customers to install Python, R, or scala libraries from standard public repositories, such as pypi, CRAN, or Maven.

Customers who are concerned about supply-chain attacks can maintain allow lists for trusted libraries within Unity Catalog.

For some deployments, customers can also host their own artifact repositories and configure Databricks to use these instead. For serverless workloads such as model serving, you can pre package dependencies that are built from your own repositories.

5.9 Use models and data from only trusted or reputable sources

Model and data supply chain attacks are growing more common, and therefore where possible organizations should only use models, weights and datasets from trusted or reputable sources such as Azure Databricks foundation models and the Azure Databricks Marketplace.

Where models or weights from untrusted sources must be used, customers should ensure that they are reviewed, scanned for malicious or vulnerable content and thoroughly tested before use. Where data from untrusted sources must be used, customers should ensure that extensive Exploratory Data Analysis has been performed.

5.10 Implement DevSecOps processes

Your data & AI code is probably the most important code base you have within your company and as such should be subject to at least the same level of scrutiny and assurance you apply elsewhere. Customers can perform static and dynamic analysis for both their code and their models.

5.11 Use lakehouse monitoring

In order to be successful with data & AI, you need to be able to have confidence in the quality of the data you're analyzing and the predictions your models are making. Azure Databricks recommends using Lakehouse Monitoring for mission critical workloads, allowing you to automatically monitor and alert on potential quality, integrity or drift issues in your data or any downstream models. Lakehouse Monitoring can also:

- Help to protect against data supply chain attacks, such as data poisoning and label flipping
- Detection PII, automatically applying tags that can be used for fine-grained access control
- Monitor fairness and bias for classification models

5.12 Use inference tables

Inference tables automatically capture incoming requests and outgoing responses to model serving endpoints and logs them to a Unity Catalog table. You can use the data in this table to monitor, debug, and improve ML models. Azure Databricks supports inference tables for LLM models in preview. Inference tables can also help to identify model inference attacks such as prompt injection, model inversion and jailbreak attempts.

5.13 Use tagging as part of your cost monitoring and charge-back strategy

To track Azure Databricks usage through to Azure Cost Management you can configure tagging on compute or pools. Tags can be combined with the billable usage system table and budgets for a 360 view of spend and subsequent chargeback.

5.14 Use budgets to monitor account spending

Budgets enable you to monitor usage across your account. You can set up budgets to either track account-wide spending, or apply filters to track the spending of specific teams, projects, or workspaces.

5.15 Use Azure Policy to create “upper limit” resource controls

While a very coarse control, Azure Policy provides an overarching control to prevent excessive resource consumption.

Appendix B - Additional Resources

Many different capabilities have been discussed in this document, with documentation links where possible. Here are some additional resources to help you learn more:

1. Review the Security and Trust Center to understand is how security built into every layer of the Azure Databricks Data Intelligence Platform, the platform architecture, the security features available and the shared responsibility model we operate under
2. Download and review the Databricks AI Security Framework (DASF) to understand how to mitigate AI security threats based on real-world attack scenarios
3. Download our due diligence package and request our Enterprise Security Guide and additional compliance reports from your Azure Databricks account team
4. Request the Azure Databricks Serverless Isolation technical guide and serverless pen test results from your Azure Databricks account team.
5. Set up the Security Analysis Tool against all workspaces, so that you can review your deployment configurations against our best practices on a continuous basis. (Learn more)
6. The foundation of good security is a robust architecture. Check out our Well Architected Framework
7. Another of the pillars of good security is strong data governance, so make sure you take a look at our Unity Catalog Best Practices
8. For more content from our security teams, please review our Platform & Security Blogs
9. If you're more of a visual person, check out our Security Best Practices YouTube series