Security Best Practices for Azure Databricks

Version 1.1.1 - June 23 2023

Azure Databricks

Table of Contents

1. Introduction

Databricks has worked with thousands of customers to securely deploy the Databricks platform with the appropriate security features to meet their architecture requirements. While many organizations deploy security differently, there are guidelines and features that are commonly used by organizations who need a high level of security.

This document details typical security features that are deployed by most high-security organizations, and reviews the largest risks and the risks that customers ask about most often. It will then provide a security configuration reference linked to our documentation and finally a collection of recommended resources.

This document is focused on the Azure Databricks platform and assumes the use of the Premium tier. It also contains some Terraform examples.

> While manually checking best practices is important when first configuring Databricks, we have produced the Security Analysis Tool (SAT) to help you monitor the security health of your Databricks Lakehouse Platform. We recommend that you set up SAT against all workspaces, so that you can review your deployment configurations against our best practices on a continuous basis. Learn more!

2. Azure Databricks architecture

Azure Databricks is a hybrid PaaS general-purpose data-agnostic compute platform.

We use the phrase hybrid PaaS because the Databricks Lakehouse 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 data plane handles your data processing. Customers deploy a data plane (virtual network and compute) in a cloud service provider account (such as AWS, Azure, or GCP) that is owned by the customer. With serverless deployments, the data plane exists in customer's Databricks account rather than their cloud service provider account. Customers get the benefits of a PaaS platform with the option to keep your data processing clusters locally within your environment.

The phrase general-purpose data-agnostic means that unlike a pure SaaS service, Databricks doesn't know what data your teams process with the Azure Databricks platform. The actual code, business logic, and the datasets are provided by your teams. You won't find recommendations like "truncate user IDs" because we don't know what data you're analyzing.

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

3. Typical security configurations

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

Importantly, these are recommendations based on the configurations we see from our customers, and following them doesn’t guarantee that you will be “secure.” Please review in the context of your overall enterprise security to determine what is required to secure your deployment and your data.

Most deployments

The following typical configurations are part of most enterprise production Azure Databricks deployments. If you are a small data science team of a few people, you may not feel the need to deploy all of these. If Databricks may become a key part of your business or if you are analyzing sensitive data, we recommend that you review these.

- ☐ Ensure that all workspaces have Secure Cluster Connectivity (No Public IP / NPIP) Enabled
- ☐ Evaluate whether multiple workspaces are required for segmentation
- ☐ Check that your storage accounts are encrypted and that public access is blocked
- ☐ Deploy Azure Databricks into your Azure virtual network (VNet injection) for increased control over the network environment. Even if you do not need this now, this option increases the chances for future success with your initial workspace
- ☐ Use IP access lists
- ☐ Use multi-factor authentication
- ☐ Separate accounts with admin privileges from day-to-day user accounts
- ☐ Run production workloads with service principals
- ☐ Configure Azure Databricks diagnostic log delivery
- ☐ Configure maximum token lifetimes for future tokens using token management
- ☐ Configure admin console settings according to your organization’s needs
- ☐ Use Unity Catalog to provide fine grained access control and centralized governance controls
- ☐ Educate users to avoid storing production datasets in DBFS
- ☐ Backup your notebooks stored in the control plane or store your notebooks in git repos
- ☐ Store and use secrets securely in Databricks or using a third-party service
- ☐ Consider whether to implement network protections for data exfiltration
- ☐ Restart clusters on a regular schedule so that the latest patches are applied

Highly-secure deployments

In addition to the configurations typical to all deployments, the following configurations are often used in highly-secure Azure Databricks deployments. While these are common, not all highly-secure environments use all of these settings. We recommend incorporating these items and the threat model in the following section alongside your existing security practices.

- ☐ Evaluate whether customer-managed encryption keys are needed on the control plane, data plane storage and data plane disks for control over data at rest (Requires Premium tier)
- ☐ Keep users and groups up-to-date using SCIM
- ☐ Use either IP access lists and/or front-end Private Link
- ☐ Configure back-end (data plane to control plane) Private Link connectivity
- ☐ Implement network protections for data exfiltration
- ☐ Use Azure Active Directory tokens for remote authentication
- ☐ Evaluate whether your datasets require blob versioning, soft deletes and other data protection strategies
- ☐ Evaluate whether your workflow requires using git repos or CI/CD
- ☐ Plan for and deploy a disaster recovery site if you have strong continuity requirements
- ☐ Encourage the use of clusters that support user isolation
- ☐ Configure cluster policies to enforce data access patterns and control costs
- ☐ Evaluate tagging to monitor and manage chargeback and cost control

4. Databricks threat model

Azure Databricks can at first appear to be an unusual architecture, but it aligns with typical PaaS security concerns. The most common threat categories that come up in customer conversations are, in priority order:

1. Account takeover or account compromise
2. Data exfiltration
3. Accidental insider exposure
4. Resource abuse such as crypto mining
5. Potential compromise of Microsoft and/or Databricks Inc.

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

Account takeover / account compromise

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

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

Mitigation strategies

1. Strongly authenticate users. The best defense to account takeover is strong authentication. Azure Databricks recommends the following best practices:
   - ☐ Leverage multi-factor authentication (MFA) with Azure AD Conditional Access
2. Implement automation that removes or disables old employee accounts when they leave your company:
   - ☐ Use SCIM for user de-provisioning and group management
3. Restrict network access. Just like other SaaS or PaaS services, Databricks does not require that users log in from a specific network (like your office or VPN) unless you enable that configuration. If an account were compromised, having network access would make it easier for an attacker to Databricks. Mitigate this risk via the following steps:
   - ☐ Ensure that all workspaces have Secure Cluster Connectivity (No Public IP / NPIP) Enabled
   - ☐ Use Azure AD conditional access
   - ☐ Use IP access lists
   - ☐ Configure private network connectivity between users and the control plane
4. Monitor user activities. Monitor user activities to detect anomalies (such as unusual time of login, or simultaneous remote logins):
   - ☐ Configure Azure Databricks diagnostic log delivery
5. Manage personal access tokens for REST API authentication. Personal access tokens are a proxy for the user who generates it, allowing full privileges. Tokens should be controlled and protected as closely as you would protect a user's credential. This is possible with:
   - ☐ Token management (recommended for most deployments)
   - ☐ Azure AD tokens (recommended for highly-secure deployments)

Data exfiltration

Risk description
If a valid user or an attacker can log into the environment, a common action on objective is to exfiltrate sensitive data from the environment, after which the attacker might store it, sell it, or ransom it.

Probability
Generally, the probability of this attack category is low because it presumes either a malicious insider or account compromise. However, if there is a malicious insider or compromised account, attackers would likely attempt to exfiltrate data.

Mitigation strategies

1. Network protections
   - ☐ Ensure that all workspaces have Secure Cluster Connectivity (No Public IP / NPIP) Enabled
   - ☐ Implement network exfiltration protections to limit where data can be sent from Azure Databricks
2. Control data access
   - ☐ Avoid storing production data in DBFS as it is accessible via API and CLI
   - ☐ Use Managed Identities to access your data and control access with storage firewalls
3. Use data exfiltration settings within the Admin Console to prevent simple methods for exfiltration

Accidental insider exposure

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

Probability
There is significant variability in the likelihood and the impact of individual errors in this category, but most security professionals identify this as a significant potential risk.

Mitigation strategies

1. Backup data and code
   - ☐ Enable blob versioning, soft deletes and other data recovery strategies
   - ☐ Backup notebooks in your environment
2. Use software development lifecycle (SDLC) processes to control what code is executed
   - ☐ Store production code in Git that you can access via the Databricks Repos feature
   - ☐ Use a CI/CD process that pushes only authorized code to production
3. Make sure your users have the access necessary
   - ☐ Use SCIM for user de-provisioning and group management
   - ☐ Monitor Azure Databricks diagnostic logs to identify what users are logging in, and what types of clusters they're configuring
   - ☐ Use clusters that support user isolation
4. Ensure that all workspaces have Secure Cluster Connectivity (No Public IP / NPIP) Enabled
5. Deploy data exfiltration protections, as the protection they provide against accidental insider exposure is similar to that provided against a malicious attacker

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.

Mitigation strategies

1. Native Azure protections
   - □ Use Azure service quotas to limit the resources that can be deployed
   - □ Regularly monitor usage data in Azure and in Azure Databricks
   - □ Regularly monitor Azure activity logs to identify abnormal provisioning activity
2. Databricks protections
   - □ Ensure that all workspaces have Secure Cluster Connectivity (No Public IP / NPIP) Enabled
   - □ Use cluster policies to limit the maximum size and type of a cluster
   - □ Limit which users can create clusters
   - □ Control the libraries that can be used in the environment to limit the risk of compromised libraries
3. Monitoring controls
   - □ Monitor utilization of your deployment with Overwatch
   - □ Monitor the Azure Databricks diagnostic logs to identify what users are logging in, and what types of clusters they're configuring

Compromise of the cloud provider and/or Databricks, Inc.

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

Probability
The security of Azure Databricks is backed by extremely strong security programs to limit the risk of an incident. However, it's never possible to fully eliminate the risk.

Databricks has an extremely strong security program which manages the risk of such an incident – see our Security and Trust Center for an overview on the program and the security features in the Databricks product. However, the risk for any company is never completely eliminated.

Azure Databricks is a Microsoft product, and the control plane is 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.

Mitigation strategies

1. Databricks controls
   - □ Ensure that all workspaces have Secure Cluster Connectivity (No Public IP / NPIP) Enabled
   - □ Monitor the Azure Databricks diagnostic log to identify the activities of Databricks employees who provide support to your deployment
   - □ Consider enabling Customer Lockbox for Microsoft Azure
2. Azure controls
   - □ Monitor Azure activity logs to identify abnormal provisioning activity
3. Process controls
   - □ Review the Databricks security documentation

4. Prepare “break-glass” customer-controls in the event of an active compromise
    - □ Disable a Databricks workspace to render applicable data unreadable by revoking a customer-managed key for managed services (not guaranteed to be a reversible operation)

5. Security configuration reference

Here are the security configurations referenced in this document.

Leverage multi-factor authentication

Azure Databricks supports Azure Active Directory conditional access, which allows administrators to control where and when users are permitted to sign in to Azure Databricks. Conditional access policies can restrict sign-in to your corporate network or can require multi-factor authentication (MFA).

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

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

Separate admin accounts from normal user 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. This best practice applies to Azure Databricks as well. If you have Databricks administrators who are also normal users of the Databricks platform (for example, there’s a lead data engineer who administers the platform and also does data engineering work), Databricks recommends creating a separate account for administrative tasks.

It’s important to note that as part of the Azure RBAC model, users that are given Contributor or above permissions to the Resource Group for a deployed Azure Databricks workspace automatically become administrators when they login to that workspace. Therefore, the same considerations outlined above should be applied to Azure portal users too.

Use service principals to run production jobs

It is against security best practices to tie production workloads to individual user accounts, and so we recommend configuring Service Principals within Databricks. Service Principles separate administrator and user actions from the workload and prevent workloads from being impacted if a user leaves an organization. With Databricks, you can configure jobs to run as service principals and generate Personal Access Tokens for Service Principals.

Use IP access lists

Configure IP access lists that restrict the IP addresses that can authenticate to Databricks by checking if the user or API client is coming from a known good 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.

As per the section above, Databricks recommends that customers combine Azure Active Directory conditional access with the use of IP access lists and/or Azure Private Link for comprehensive control over which networks their workspaces can be accessed from.

IP access lists can be configured by using Terraform:

```
data "http" "my" {
  url = "https://ifconfig.me"
}

resource "databricks_workspace_conf" "this" {
  custom_config = {
    "enableIpAccessLists": true
  }
}

resource "databricks_ip_access_list" "only_me" {
  label = "only ${data.http.my.body} is allowed to access workspace"
  list_type = "ALLOW"
  ip_addresses = ["${data.http.my.body}/32"]
  depends_on = [databricks_workspace_conf.this]
}
```

This feature requires the Premium pricing tier.

Configure Azure Private Link

Azure Private Link provides a private network route from one Azure environment to another. Private Link can be configured both between Azure Databricks users and the control plane, and also between the control plane and the data plane.

Between Databricks users and the control plane, Private Link provides strong controls that limit the source for inbound requests. If a company already routes traffic through an Azure environment, they can use Private Link so that the communication between users and the Azure Databricks control plane does not traverse public IP addresses.

This feature requires the Premium pricing tier.

Azure Databricks diagnostic log delivery

Azure Databricks provides comprehensive end-to-end diagnostic logs of activities performed by Azure Databricks users, allowing you to monitor user activities to detect anomalies such as unusual time of login or simultaneous remote logins. For an example of this, Databricks recommends that customers review the blog post *monitoring your Databricks Lakehouse platform with audit logs*. This post shows common detections and provides a working data pipeline to monitor the Databricks diagnostic log, although Azure Databricks customers often prefer to analyze their logs in an Azure Log Analytics workspace instead. Customers can create Azure Monitor alerts to notify them of specific events based on the Log Analytics queries that they write.

This feature requires the Premium pricing tier.

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. Highly-secure customers typically provision a maximum token lifetime for new tokens for a workspace.

This feature requires the Premium pricing tier.

Use Azure Active Directory tokens for remote authentication

AAD tokens are time limited based on configuration based on your AAD tenant. They're also fully integrated into Azure AD and therefore visible throughout the Azure IAM monitoring stack. Where possible, customers should authenticate using Azure Active Directory tokens rather than using Azure Databricks PATs. AAD tokens are strongly recommended for workspace and infrastructure automation, particularly where the AAD token is generated on behalf of a service principal.

This feature requires the Premium pricing tier.

Ensure that all workspaces have Secure Cluster Connectivity (No Public IP / NPIP) Enabled

With secure cluster connectivity enabled, customer virtual networks have no inbound open ports from external networks and Databricks cluster nodes have no public IP addresses. Secure cluster connectivity is also known as No Public IP (NPIP). Databricks recommends this configuration for all Azure Databricks workspaces because it significantly reduces the attack surface and hardens the security posture.

Deploy Azure Databricks in your own Azure virtual network (VNet injection)

The default deployment of Azure Databricks is a fully managed service on Azure: all data plane resources, including a VNet that all clusters will be associated with, are deployed to a locked resource group. If you require network customization, however, you can deploy Azure Databricks data plane resources in your own virtual network, and Databricks recommends this approach for nearly all deployments. Even if you do not need network customisation now, you may do in the future, and this option increases the chances for future success with your initial workspace.

When designing your virtual network, you also need to consider your egress architecture. An Azure NAT gateway is recommended for most deployments, but if data exfiltration protection is required you may need to consider a different egress architecture.

Important note: while it is possible to deploy Databricks with public IP addresses, Databricks recommends our Secure cluster connectivity (No Public IP / NPIP) model and it is very unusual for a security-conscious customer to prefer public IPs.

Implement network exfiltration protections

By default, Azure Databricks data plane hosts within your Azure subscription have unlimited outbound network access. If you have deployed Azure Databricks in your own virtual network, you can lock down outbound access. Databricks has created a blog post that discusses how to do this using an Azure Firewall, but it can be generalized to other network security tools using details in Azure Databricks documentation.

The TLS connections between the control plane and the data 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.

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 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. Databricks recommends that our customers do not store production data in DBFS.

Evaluate Private Link to access your Azure resources

While most customers will not use Azure Private Link to connect from Azure Databricks to their Azure storage due to the cost, it is used where regulations or organizational policies require that all data access traverses only private IP addresses. Use storage firewalls to restrict access to your trusted private endpoints, networks and managed identities.

Access Storage via Managed Identities

Unity Catalog allows you to use Azure managed identities to access storage. Some of the major benefits of using Managed Identities are:

- You don't need to manage credentials. Credentials aren’t even accessible to you.
- You can use managed identities to authenticate to any resource that supports Azure AD authentication.
- Managed identities can be used at no extra cost.

Databricks recommends using Unity Catalog and Managed Identities 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 data 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 for more details.

Leverage data exfiltration settings within the admin console

The admin console contains a variety of settings that provide protection. Most admin console controls are simple enable/disable buttons. Some of the most important ones are:

☐ Export notebooks or cells containing code and partial interactive query results
☐ Download notebook results
☐ Block notebook clipboard features
☐ Download MLflow run artifacts
☐ Block application attacks via iFrames and cross-site scripting

Enable blob versioning, 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. Consider the various options available and apply as necessary to achieve your target Recovery point objective (RPO):

- Resource locks
- Blob versioning
- Soft deletes for containers
- Soft deletes for blobs
- Storage redundancy
- Delta clones

Monitor workspace using Overwatch

Overwatch was built to enable Databricks' customers, employees, and partners to quickly / easily understand operations within Databricks deployments. As enterprise adoption increases there’s an ever-growing need for strong governance. Overwatch means to enable users to quickly answer questions and then drill down to make effective operational changes.

Most use cases are not strictly security-focused, but improved visibility strengthens all security teams. For example, suppose a PyPi library you incorporate is compromised by crypto miners, you would be grateful to have a tool that excels at troubleshooting heavy utilization within your Databricks deployment.

Backup via the Databricks migration tool

The Databricks migration tool allows admins to export most portions of their workspace and then import them into a new workspace. It is often used as a part of disaster recovery strategies to perform batch analytics of notebook code (such as secret hunting) or as a general-purpose backup of code stored in the Databricks control plane.

Guidance around these tools:

- migrate is a tool to migrate a workspace one time. It uses the Databricks CLI/API in the background.
- databricks-sync is a tool that has been used for multi cloud migrations, as well as disaster recovery synchronization of workspaces. It uses the Terraform provider to synchronize incremental changes.
- You can run either tool from a command line or from a notebook.

Store code in Git

The Databricks Repos feature allows you to move away from data stored just in the Azure Databricks control plane, and instead store data in a Git repo. You can check code in/out and switch branches. Use Repos for better code visibility and tracking.

Manage code run via CI/CD

Mature organizations often build production workloads by using CI/CD to integrate code scanning, better control permissions, 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. The Azure Databricks documentation provides detailed examples of CI/CD using:

- Azure DevOps
- Jenkins

Configure a disaster recovery (DR) site

While Databricks doesn’t offer disaster recovery services, many customers use Databricks capabilities including the Account API to create a cold (on standby) workspace in another region. We have documentation to guide customers through this process.

Isolate sensitive workloads into different workspaces

While Azure Databricks has numerous capabilities for isolating different workloads, for very sensitive workloads, the best unit of isolation might be to move them 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 different data in Databricks.

SCIM synchronization of users and groups

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

1. When you remove a user, the user is automatically removed from 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

Please refer to the documentation for detailed instructions on how to configure SCIM for Azure Databricks.

This feature requires the Premium pricing tier.

Encrypt storage accounts and restrict access

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. The former is protected by Azure Deny Assignments, although customers should consider whether they want to encrypt it with a customer-managed key rather than the default Azure managed key. Customers can also choose to additionally encrypt it at the infrastructure level with double encryption.

For the storage accounts that you manage, it is your responsibility to ensure that the storage accounts are protected according to your requirements. Examples might include:

- Encryption with your customer-managed key
- Restrict access to trusted networks with a storage firewall
- Anonymous public access is not allowed

See the Azure security baseline for Storage for an exhaustive list.

Add a customer-managed key for managed services

Add a customer-managed key for select data stored within the Azure Databricks control plane, such as notebooks, secrets, Databricks SQL queries, and Databricks SQL query history.

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.

This feature requires the Premium pricing tier.

Add a customer-managed key for DBFS root

Add a customer-managed key for the root storage account used for DBFS. 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.

This feature is only available at the Premium pricing tier.

Add a customer-managed key for managed disks

Add a customer-managed key for the managed disks that are attached to a cluster. 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.

This feature is only available at the Premium pricing tier.

Monitor provisioning activities in Azure Monitor

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. The Azure Databricks diagnostic log is 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.

Cloud provider audit logs such as the Azure Monitor activity log provide a great mechanism for observing the behavior of Azure Databricks in the data plane. It provides visibility into:

- Virtual Machine (VM) creation, to help identify bitcoin mining and also control for billing.
- Subscription level events such as API calls, to help identify account compromise.

These activity logs can be joined to resource logs, such as the Azure Databricks diagnostic logs, NSG flow and other network logs and Azure Active Directory logs for a 360 degree view of what's happening within your Azure account, as well as providing a historical baseline of what "normal behavior" looks like. See the Azure documentation for more information.

Customer Lockbox for Microsoft Azure

By default, Databricks personnel do not have access to customer workspaces or to the production multi-tenant environments. Databricks staff may request temporary access to a customer workspace in order to investigate an outage, security event, or to support your deployment. In the rare circumstances where such access is required, Customer Lockbox for Microsoft Azure provides an interface for customers to review and approve or reject customer data access requests.

Please refer to the documentation for more details about Customer Lockbox for Microsoft Azure.

Azure service quotas

While a very coarse control, Azure service quotas provide an overarching control to prevent excessive resource consumption.

Use Unity Catalog to enhance data governance

Unity Catalog is a unified data product catalog designed to simplify governance of data and AI assets on the lakehouse. It brings fine-grained, centralized governance to data assets (tables, files, dashboards, ML models, etc.) through a simple, standard interface (ANSI SQL) that works with existing data across clouds and storage systems.

Data teams are often stuck defining permissions at a coarse grained level. Unity catalog provides a modern approach to granular access control with centralized policy, auditing, and lineage tracking, all integrated into your Azure Databricks workflow.

Unity Catalog enables:

- Security or governance teams to centrally manage and govern data assets
- Manage fine-grained access controls with ease
- Unify and secure the data search experience
- Enhance query performance at any scale
- Automate real-time data lineage
- Securely share data across organizations with Delta Sharing

Use clusters that support user isolation

When Databricks started, “No Isolation Shared” clusters (formerly called “standard” mode or “standard clusters”) were the default because our customers were small engineering teams who had the same access to the same datasets. But over the years, we built out new user isolation capabilities that meet the security needs of our customers today, and so recommend utilizing clusters that support user isolation if possible.

The following types of clusters will enforce user isolation so that users with different privilege levels can coexist on the same cluster:

- SQL Warehouses
- Any cluster that does not have access mode set to “No isolation shared”

If you are using the legacy Manage Clusters UI within the data science or data engineering workspaces, the following cluster types will enforce user isolation similarly to “No Isolation Shared” clusters:

- High concurrency clusters with table access control lists (Table ACLs clusters for short)
- High concurrency clusters with credential passthrough

Clusters with user isolation include enforcement such that each user runs as a different non-privileged user account on the cluster host. Languages are also limited to those that can be implemented in an isolated manner (SQL and Python), and Spark APIs must be on an allowlist of those we believe to be isolation-safe.

SQL Warehouses also enforce user isolation and have similar safety features, though implemented in a mechanism specific to the SQL workloads run on these clusters.

Customers with more stringent security requirements can enforce cluster policies that do not allow standard clusters to be created within the environment.

If you need standard clusters, customers whose workspaces have Unity Catalog enabled can allow users to create single-user clusters. Administrators for workspaces without unity catalog can create clusters and use Cluster ACLs to control the users permitted to attach notebooks.

Cluster policies

Azure Databricks administrators can control many aspects of the clusters that are spun up, including available instance types, Databricks versions, and the size of instances by using cluster policies. Admins can enforce some Spark configuration settings. Admins can configure multiple cluster 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. For detailed recommendations and discussion of cluster policies, see our announcement blog post.

Limiting cluster creation rights

Using either cluster policies or the older cluster ACLs, admins can define what users or groups within the organization are able to create clusters.

Cluster ACLs allow you to specify which users can attach a notebook to a given cluster. Note that if a user shares a notebook already attached to a standard mode cluster, the recipient will also be able to execute code on that cluster. This does not apply to clusters that enforce user isolation: SQL Warehouses, high concurrency with table ACLs clusters, and high concurrency with credential passthrough clusters. Customers who use Unity Catalog can also enable single-user clusters to enforce isolation clusters.

Controlling libraries

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

Those who are concerned about supply-chain attacks, can host their own repositories and then configure Azure Databricks to use those instead. You can block access to other sources of libraries. Documentation for doing so is outside the scope of this document, but reach out to your Databricks team for assistance as required.

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 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 controls 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.

Configure tagging to monitor usage and enable charge-back

To track Azure Databricks usage through to Azure billing, you can configure tagging on workspaces, clusters or pools. These can also be enforced via cluster policies for different groups within your organization.

Restart clusters on a regular schedule

Azure Databricks 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. Serverless customer workloads are limited to a maximum of 7 days of total uptime, so serverless compute cannot go without updates in the way that is possible for the classic data plane.

6. Resources

Many different capabilities have been discussed in this document, with documentation links where possible. Organizations who prioritize high security can learn more than what is in this document. Here are additional resources to help you learn more:

1. 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)
2. Request the *Enterprise Security Guide* and compliance documentation from your Databricks account team.
3. Review the security features in the Security and Trust Center, along with the overall documentation about the Databricks security and compliance programs.
4. The Security and Trust Overview Whitepaper provides an overview of the Databricks architecture and platform security practices.
5. Documentation articles:
    a. Security guide - Azure Databricks
    b. Azure security baseline for Azure Databricks
    c. Enterprise security for Azure Databricks

6. Blog: Azure Databricks Security Best Practices
    a. See also our blog on Data Exfiltration Protection with Azure Databricks
7. Whitepaper: Data Plane Host Security Summary (request from your Databricks Account Team)
8. Documentation article: Customer support access, including customer-approved workspace login

7. More Information

For more information about Azure Databricks security, please see the Enterprise Security Guide and Azure Databricks Docs. Your Azure Databricks representatives will be happy to assist you with copies of these documents.