AZURE BLOCKLIST INTEGRATION

## Prerequisites

Before proceeding with the integration, ensure the following components are already set up in your Azure environment:

* **API Access**: Service principal or user credentials with sufficient permissions to read and modify relevant Azure resources.
* **Microsoft.Web**: The **Microsoft.Web** resource provider must be registered (used for Azure App Services and related components).
* **Microsoft.Insights**: The **Microsoft.Insights** resource provider must be registered (used for monitoring and diagnostics, such as Application Insights or Log Analytics).
* Ensure your firewall policy does **not** already contain a Rule Collection Group with a **priority of 100**.

## Required Variables

To successfully configure the integration, gather the following details from both your **Azure environment** and your **Celerium account**:

**Azure Environment:**

* Subscription Name or ID
* Resource Group Name
* Location
* Function App Name
* Firewall Name
* Firewall Policy Name
* Tenant ID
* Client ID
* Client Secret

**Celerium Account:**

* Celerium Blocklist URL

For additional configuration options and optional variables, refer to the [readme file](https://github.com/Celerium-Inc/AzureNetworkFirewallSupport/blob/main/Blocklist%20Integration/README.md) provided in the GitHub repository.

## Data Retrieval

Before deploying the integration, you need to collect specific information from both your **Azure environment** and the **Celerium portal**. Follow the steps below:

* **Subscription Name or ID**

To locate your Azure subscription details:

1. In the **Azure Portal**, navigate to **Subscriptions**.
2. Ensure that no filters (e.g., directories or tags) are applied so all subscriptions are visible.
   * + A screenshot of a chat

       AI-generated content may be incorrect.
   1. Identify and **note** the **Subscription Name** or **Subscription ID** you intend to use for the integration.

* **Resource Group Name**
  1. In the **Azure Portal**, go to **Resource groups**.
  2. Locate and **note** the name of the **Resource Group** where your resources (e.g., Function App, Firewall) are or will be deployed.
* **Location**
  1. In the **Azure Portal**, open the **Resource Group** you selected earlier.
  2. The **Location** (Azure region) will be listed at the top of the overview pane (e.g., eastus, westeurope).
  3. **Note** the location for use in your deployment.
* **Function App Name**
  1. Choose a name for your **Function App** that meets the following Azure naming requirements:
     + Must be **1 to 60 characters** long
     + Can include **lowercase letters**, **numbers**, and **hyphens**
     + Must **start and end** with a letter or number
     + Cannot contain **consecutive hyphens**
     + Must be **globally unique** across all of Azure
  2. **Suggested name**: celerium-blocklist
  3. **Note** the Function App name you’ve selected for use during deployment.
* **Firewall Name**
  1. In the **Azure Portal**, search for **Firewalls** in the top search bar.
  2. From the list, identify and **note** the name of the **Azure Firewall** you intend to use.
* **Firewall Policy Name**
  1. In the **Azure Portal**, go to **Firewalls** and select the firewall you're using.
  2. From the **Overview** tab, locate the **Firewall policy** section. The policy name will be displayed there.
     + A screenshot of a computer

       AI-generated content may be incorrect.
  3. **Note** the name of the **Firewall Policy** for use in the integration.
* **Tenant ID and Client ID**
  1. In the **Azure Portal**, search for **Microsoft Entra ID** (formerly Azure Active Directory).
  2. Navigate to **Manage > App registrations > All applications**.
  3. Click on the name of the application used for **API access**.
     + A screenshot of a computer

       AI-generated content may be incorrect.
  4. On the **Overview** page:
  + **Note** the **Client ID** (also labeled as *Application (client) ID*).
  + **Note** the **Tenant ID** (also labeled as *Directory (tenant) ID*).
* **Client Secret**
  1. **Client secrets cannot be retrieved** once created in Azure. If you no longer have access to the existing secret value, it cannot be recovered.
  2. If you have stored the **client secret** elsewhere (e.g., in a password manager or documentation), **note** it for use in the integration.
  3. If the secret is unavailable, you will need to **generate a new client secret** or create a new **App Registration** for API access.
* **Celerium Blocklist URL**
  1. Log in to the **Celerium portal**.
  2. Locate your assigned **blocklist URL** in the portal dashboard or settings.
  3. **Note** the URL for use in the integration.

## Preparing for Deployment

There are several ways to deploy this integration, such as:

* **Azure Cloud Shell** (recommended – no local installation required)
* **Local machine** with the Azure PowerShell module installed
* **Visual Studio Code** with Azure extensions

This guide uses **Azure Cloud Shell**, which is accessible directly from the Azure Portal.

**Accessing Azure Cloud Shell**

You can access **Azure Cloud Shell** directly from the Azure Portal by clicking the **Cloud Shell** icon in the top navigation bar. Cloud Shell supports both **PowerShell** and **Bash** environments.

**Browser Compatibility Recommendation:**  
For the most stable experience with Azure Cloud Shell:

* Use the **Firefox** browser, which has shown the **least number of connectivity and rendering issues** during deployment and scripting workflows.
* While **Chrome** and **Edge** are supported, Firefox performs better with shell responsiveness and file uploads.

**Troubleshooting Tip:**  
If you encounter issues like:

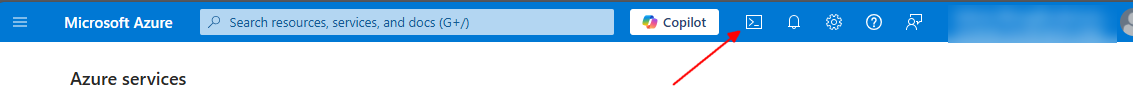
* Cloud Shell becoming **unresponsive**
* The **prompt not scrolling** or **overwriting previous lines**

**Try the following:**

1. Click **Restart** in the Cloud Shell window.
2. If the issue persists, **close and reopen** Cloud Shell from the portal.

**Getting Started with Azure Cloud Shell**

1. Log in to your **Azure Portal** at <https://portal.azure.com>.
2. Click the **Cloud Shell icon** (command prompt symbol) in the top-right of the portal, as shown below:



1. A **PowerShell prompt (PS>)** will appear at the bottom of your browser window.

A screenshot of a computer

AI-generated content may be incorrect.

* If Bash is shown, click **"Switch to PowerShell"** in the upper-left corner.

A screen shot of a computer

AI-generated content may be incorrect.

**Prepare Your Environment**

1. Have the **environment variables** you gathered earlier on hand — you’ll need them during deployment.
2. Download the integration files from **GitHub** to your local machine.

* **Important**: Maintain the **original folder structure** as shown in the GitHub repository.

**Set the Correct Azure Subscription**

1. Verify that you're in the correct subscription before proceeding:

* To **check your current subscription**:
* **Get-AzContext**

A blue screen with green text

AI-generated content may be incorrect.

1. To **switch subscriptions**, use either the name or the ID:

* By name
  + **Set-AzContext -Subscription "your subscription name"**
* By ID
  + **Set-AzContext -SubscriptionId "your-subscription-guid-here"**
* A screenshot of a computer program

  AI-generated content may be incorrect.

1. Confirm the subscription was updated:

* Press the **up arrow twice** to recall **Get-AzContext**, then **press Enter**.
* Confirm the correct subscription is now active.

**Upload Integration Files**

1. Upload the necessary files from GitHub:

* In the Cloud Shell window, click **Manage Files > Upload**.
* A screenshot of a computer

  AI-generated content may be incorrect.
* Upload **only the files from the src directory** of the **Blocklist Integration**.
* These files will be placed in the /home/admin/ directory by default.

1. Create the required folder structure:

**mkdir ./block**

**mkdir ./block/src**

1. Move the uploaded src files into the new folder:

**move-item -path "/home/admin/\*.\*" -destination "/home/admin/block/src"**

1. Upload the **cleanup.ps1** and **deploy.ps1** scripts from the root of the **Blocklist Integration** directory.
2. Move those scripts to the block directory:

**move-item -path "/home/admin/\*.\*" -destination "/home/admin/block"**

## Deploying Blocklist

Follow the steps below to deploy the Blocklist integration using your prepared environment variables and the deployment script.

**Prepare the Deployment Command**

Using the variables you gathered earlier, build your deployment command in a text editor (e.g., Notepad). Use the format below as a reference.

Example for :

**./block/deploy.ps1 `**

**-ResourceGroupName "your-rg" `**

**-Location "your-loc" `**

**-FunctionAppName "your-func-name" `**

**-FirewallPolicyName "your-policy" `**

**-FirewallName "your-firewall" `**

**-TenantId "your-tenant-id" `**

**-ClientId "your-client-id" `**

**-ClientSecret "your-client-secret" `**

**-BlocklistUrl** [**https://your-blocklist-url**](https://your-blocklist-url)

For more information and other optional variables (i.e. Gov cloud environment), see the [readme file](https://github.com/Celerium-Inc/AzureNetworkFirewallSupport/blob/main/Blocklist%20Integration/README.md) on GitHub.

**Run the Deployment Script**

1. **Copy** the command you created and **paste it into the Azure Cloud Shell PowerShell prompt**.

A blue screen with white text

AI-generated content may be incorrect.

1. When prompted, enter **Y** to confirm and proceed.

**Review Script Output**

1. After initial execution, you will see progress messages.

A blue screen with a blue background

AI-generated content may be incorrect.

1. You may be prompted again — enter **Y** to continue.

**Final Deployment Step**

1. Once the Azure resources are deployed, you'll see additional output.

A screenshot of a computer program

AI-generated content may be incorrect.

1. A final prompt will ask to restart the Function App — enter **Y**.

You should now see a confirmation message indicating that the deployment is complete.

A blue screen with white text

AI-generated content may be incorrect.

**Note:** Although deployment finishes, the Azure Function App may take up to **15 minutes** to finish building the IP blocklists and updating the Firewall Policy.

## Post-Deployment

After deploying the Blocklist Function App, follow the steps below to verify it is working correctly and that your Azure Firewall Policy has been updated accordingly.

### Monitor the Function App

1. In the **Azure Portal**, search for **Function App**.
2. Click on the **Function App** you created during deployment.
3. In the left-hand menu, click on the **function** named **blocklist**.

A screenshot of a computer

AI-generated content may be incorrect.

If you don’t see the blocklist function as shown above and see an error message as shown below, you can click on either Refresh buttons as indicated in the screenshot.

A screenshot of a computer

AI-generated content may be incorrect.

If you clicked on blocklist and received the error while loading as shown below, click on Refresh.

A screenshot of a computer

AI-generated content may be incorrect.

1. Click the **Logs** tab

* The log should show that the function has **connected and begun processing**.
  + A screenshot of a computer

    AI-generated content may be incorrect.
* Depending on the number of IPs retrieved, output may take up to **15 minutes** to appear.

### Check IP Groups

Once the function has run successfully, it will create **IP Groups** used in your Firewall Policy.

* In the Azure Portal, search for **IP Groups**.
* Look for IP groups with names starting with:

“**fw-blocklist-###**“

* + A screenshot of a computer

    AI-generated content may be incorrect.
* The number of IP Groups created depends on how many IPs are in the blocklist.
  + For more details, refer to the [readme file](https://github.com/Celerium-Inc/AzureNetworkFirewallSupport/blob/main/Blocklist%20Integration/README.md) on GitHub.

### Check Firewall Policy

The function creates a new **Rule Collection Group** within your existing firewall policy.

1. In the Azure Portal, search for **Firewall Policies**.
2. Click on the firewall policy name you noted earlier.
3. Navigate to **Rules > Rule collections**.

* You should see a rule collection group named:

“**CeleriumRuleCollectionGroup**”

A screenshot of a computer

AI-generated content may be incorrect.

* This group contains:
  + A **Network Rule Collection** named **Blocked-IP**, which includes one rule to **block outbound traffic** to the blocklisted IPs.
  + A **DNAT Rule Collection** named **Blackhole**, which is currently **empty** — this is where you'll add **inbound block rules** using the workaround below.

## Blocklist with Inbound Traffic (Manual Workaround)

Azure Firewall does **not natively support** denying inbound traffic using IP blocklists. However, a **manual workaround** can effectively block traffic from malicious IPs.

This approach involves **duplicating your existing DNAT rules** and redirecting traffic from blocklisted IPs to a **non-existent internal IP address**, effectively “blackholing” the traffic.

**How the Workaround Works**

For each DNAT rule:

* Set **source** to your **IP Groups** (i.e., the blocklists)
* Set the **translated address** to a **non-existent internal IP** (e.g., 10.0.255.254)

Since there's no backend server at that address, Azure Firewall **silently drops** the traffic.

**Example:**

**Original DNAT Rule (Legitimate Access)**

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | SFTP Access |
| Source Type | IP Address |
| Source | \* (any) |
| Protocol | TCP |
| Destination Ports | 40888 |
| Destination Address | Public IP associated with Firewall |
| Translated Type | IP Address |
| Translated Address | 10.11.2.4 (actual server IP) |
| Translated Port | 40888 |

**A close-up of a document

AI-generated content may be incorrect.**

**Blackhole Rule (Deny Blocklisted IPs)**

|  |  |
| --- | --- |
| **Field** | **Value** |
| Name | BlackholeSFTPAccess |
| Source Type | IP Group |
| Source | **fw-blocklist-001, fw-blocklist-002, etc.** |
| Protocol | TCP |
| Destination Ports | 40888 |
| Destination Address | Same public IP as original rule |
| Translated Type | IP Address |
| Translated Address | **10.11.255.254 (non-existent)** |
| Translated Port | 40888 |

**Creating the Blackhole DNAT Rule**

1. In the Azure Portal, go to **Firewall Policies**.
2. Click on the firewall policy name you have noted earlier.
3. Navigate to **Rules > DNAT Rules**.
4. Click on **+ Add rule**.
5. Set **Rule collection group** to **CeleriumRuleCollectionGroup**.
6. Set **Rule collection** to **Blackhole.**
7. Name the rule (e.g. BlackholeSFTPAccess).
8. Set **Source Type** to **IP Group**
9. Under **Source IP Groups**, select all IP groups starting with **fw-blocklist**.
10. Set **Destination Address**, the **same public IP** as the original DNAT rule.
11. Set **Protocol** to match the original rule (e.g. TCP).
12. Set **Destination Ports** to the same as the original rule (e.g. 40888).
13. Set **Translated Type** to **IP address**.
14. Set **Translated Address** to a **non-existent IP** (e.g. 10.11.255.254).
15. Set **Translated Port** to match the original (e.g. 40888).
16. Click **Save**.

A screenshot of a computer

AI-generated content may be incorrect.

After Rule creation:

A screenshot of a computer

AI-generated content may be incorrect.

**Repeat as Needed**

You must **repeat this process for** **each inbound DNAT rule**.

Every legitimate DNAT rule should have a **matching "blackhole" rule** using the blocklist IP groups as the source.

**Important Limitation:**  
Azure Firewall supports a maximum of **250 DNAT rules**, calculated as the number of firewall public IP addresses plus unique destination combinations (destination address, port, and protocol). Since Azure Firewall does **not allow combining multiple destination IPs or ports in a single rule**, each unique destination requires a separate DNAT rule.

If the blackhole workaround is implemented, the effective limit is **125 DNAT rules**, because each original DNAT rule is duplicated to create its corresponding blackhole rule, reaching the maximum of 250.

## View Traffic Logs That Were Blocked

### Enabling Firewall Logs for Blocked Traffic

1. In the Azure portal, navigate to your **Firewall** resource.
2. Select **Monitoring → Diagnostic settings** from the left-hand menu.
3. Click **+ Add diagnostic setting**.
4. Enter a descriptive name, e.g. **NAT\_Logs**.
5. Under **Logs → Categories**, check **Azure Firewall NAT Rule**.
6. Under **Destination details**, select **Send to Log Analytics workspace**.
7. Choose the correct **Subscription** and **Log Analytics Workspace (LAW)**.
8. For **Destination table**, select **AzureDiagnostics**.
9. Save the configuration.

**Note:** After enabling Azure Firewall diagnostic logging, it can take **several hours** for data to begin appearing in your Log Analytics workspace.

A screenshot of a computer

AI-generated content may be incorrect.

### Running the Query to View Blocked Traffic

1. In the Azure portal, open your **Log Analytics workspace (LAW)**.
2. Select **Logs** from the left-hand menu.
3. If the Queries hub opens, close it.
4. In the query editor, switch from **Simple mode** to **KQL mode** (top right).
5. Set an appropriate **Time Range** for the query.
6. Paste and run the following query:

AzureDiagnostics

| where Category == "AZFWNatRule"

| where RuleCollectionGroup\_s == "CeleriumRuleCollectionGroup" // inbound and outbound

//| where RuleCollection\_s == "Blackhole" // for inbound only

//| where RuleCollection\_s == "Blocked-IP" // for outbound only

| project TimeGenerated, SourceIP, DestinationIp\_s, Protocol\_s, RuleCollection\_s, Rule\_s

| order by TimeGenerated desc

By default, the query above returns **both inbound and outbound blocked traffic.**

To view **inbound traffic only**, uncomment the line (by removing the // at the beginning of the line):

| where RuleCollection\_s == "Blackhole" // for inbound only

To view **outbound traffic only**, uncomment the line:

| where RuleCollection\_s == "Blocked-IP" // for outbound only

Use // to comment out any line you do **not** want applied to the query.

### Saving the Query for Reuse

1. After running the query, click **Save → Save as query** at the top of the Logs blade.
2. Give the query a descriptive name, such as **Blocked Traffic Report**.
3. Fill out the dialog box and click **Save.**

## Removing the Function App and Blocklists

### Removing the Function App

Before starting, ensure you've completed the **"Prepare for Deployment"** section of this guide.

1. Open a text editor (e.g., **Notepad**) and prepare your PowerShell command using the example below. For more details, refer to the **README** file in the GitHub repository.
2. Example command:

./block/cleanup.ps1 `

-ResourceGroupName "your-rg" `

-FunctionAppName "your-func-name"

For more information and other optional variables (i.e. Gov cloud environment), see the [readme file](https://github.com/Celerium-Inc/AzureNetworkFirewallSupport/blob/main/Blocklist%20Integration/README.md) on GitHub.

1. Copy the command with your actual values and paste it into **Azure Cloud Shell**.

A screen shot of a computer screen

AI-generated content may be incorrect.

* + When prompted, enter **Y** to confirm the deletion.

1. Once the function app is removed, you should see confirmation output similar to:
   * A blue screen with white text

     AI-generated content may be incorrect.

### Removing the Blocklists

**Step 1: Delete Rule Collection Group**

1. Go to the **Azure Portal** and search for **Firewall Policies**.
2. Select the firewall policy you used during deployment.
3. Navigate to **Rules > Rule collections**.
4. Locate the line containing **CeleriumRuleCollectionGroup**.
5. Click the **⋮ (three-dot menu)** on the far right and select **Delete**.
   * A screenshot of a computer

     AI-generated content may be incorrect.
6. Monitor progress using the **notification bell** at the top of the portal.
   * Note: Deletion can take a few minutes. Wait until it fully completes.
     + A screenshot of a computer

       AI-generated content may be incorrect.

**Step 2: Delete IP Groups**

1. In the Azure Portal, go to **All resources**.
2. In the filter bar, enter:

**fw-blocklist**

1. Make sure you're filtering by the correct **subscription**.
2. Select all IP Group resources found by checking the box next to **Name**.
3. Click **Delete.**
   1. A screenshot of a computer

      AI-generated content may be incorrect.
4. A confirmation pane will appear summarizing what will be deleted.
   1. A screenshot of a computer

      AI-generated content may be incorrect.
5. **Verify** that this list is accurate.
6. At the bottom, type Delete, then click **Delete** again.

A screenshot of a computer error

AI-generated content may be incorrect.

1. A final confirmation box may appear — confirm it.

A screenshot of a computer error

AI-generated content may be incorrect.

**Result**

You have now fully removed:

* The Function App
* Associated IP Groups
* The Firewall Rule Collection Group

This completes the teardown of the blocklist integration.