

 Featured

RED TEAM SERIES

Red Teaming in Cloud: Leverage Azure FrontDoor CDN for C2 Redirectors



Nairuz Abulhul · [Follow](#)

Published in R3d Buck3T · 10 min read · Jan 11, 2024

 20



Evading Detection: Obfuscating C2 Infrastructure with Azure FrontDoor



Photo by [Yifu Wu](#) on [Unsplash](#)

A redirector is a server that acts as a middleman between the C2 server and the targeted network. Its primary function is redirecting all communication between the C2 and the compromised target. Redirectors are commonly used to hide the origin of the traffic of the C2 server, making it more challenging for defenders to detect and block the C2 infrastructure.

Cloud-based redirectors present a good opportunity to obscure the C2 traffic by routing it through a global network of servers such as the Content

Delivery Network (CDN). They are simple to set up, and if a C2 channel is detected, the red team can quickly create a new redirector instead of reconstructing the entire infrastructure.

In this blog post, we will continue our [Red Team series](#). We'll discuss the Azure FrontDoor CDN service and how it can be utilized as a redirector for our C2 infrastructure.

• • •

Table of Content

- [Redirector Setup](#)
- [Prerequisites](#)
 - [Enabling Microsoft.CDN Provider \(Azure Trial\)](#)
 - [Configuring VM Firewall Rules](#)
- [Configuring Azure Front Door CDN Endpoint](#)
- [Accessing the Endpoint](#)
 - [Routing Options Configuration](#)
 - [Origin groups Configuration](#)
 - [CDN Endpoint](#)
- [Resources](#)

• • •

Redirector Setup

The setup involves a CDN redirector that communicates with a targeted system via HTTPS on port 443 and between the CDN and the C2 server via HTTP on port 80, as shown in the diagram.

This setup ensures that any callback from a compromised system will not directly go to the C2 server, thus hiding the actual location of the C2 server.

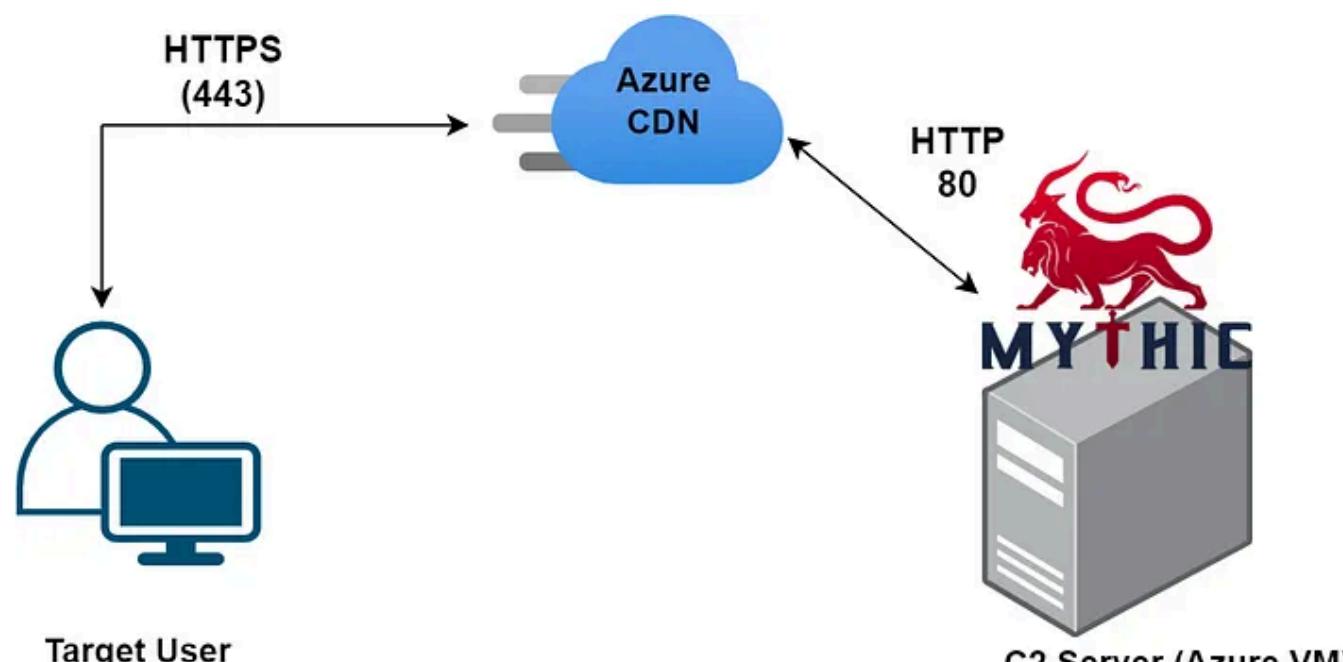


Figure 01 — shows the connection between the target user, Azure CDN, and the C2 server.

Prerequisites

This section outlines the key steps required to set up CDN redirectors. These steps include enabling the *Microsoft.CDN* Provider and configuring VM Firewall Rules.

If you need to get a better understanding of the Azure environment, you can refer to the first article in this guide titled [“Red Teaming in the Cloud: Deploying Azure VMs for C2 Infrastructure.”](#)

Enabling Microsoft.CDN Provider (Azure Trial)

If you are currently using the Azure Free trial service and want to create a CDN profile, you must enable the “Microsoft.CDN” provider.

To do this, search for your subscription name and select it. From there, navigate to the left menu and select “Resource providers” under *Settings*.

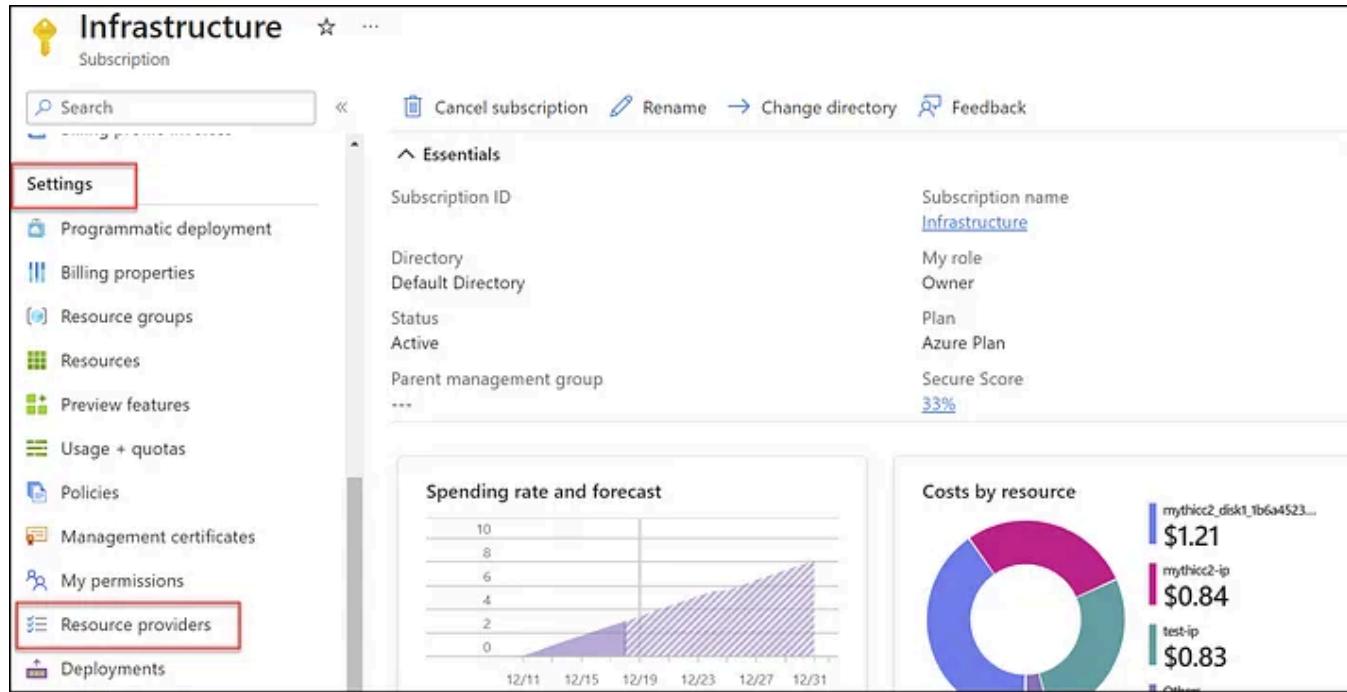


Figure 02 — shows the location of the Resource Providers in the Subscription settings.

In the “Resource Providers” section, search for *Microsoft.Cdn*. If the provider is not registered, you will see the status as “Not Registered.” To register the provider, click on *Microsoft.Cdn* and then click the Register button.

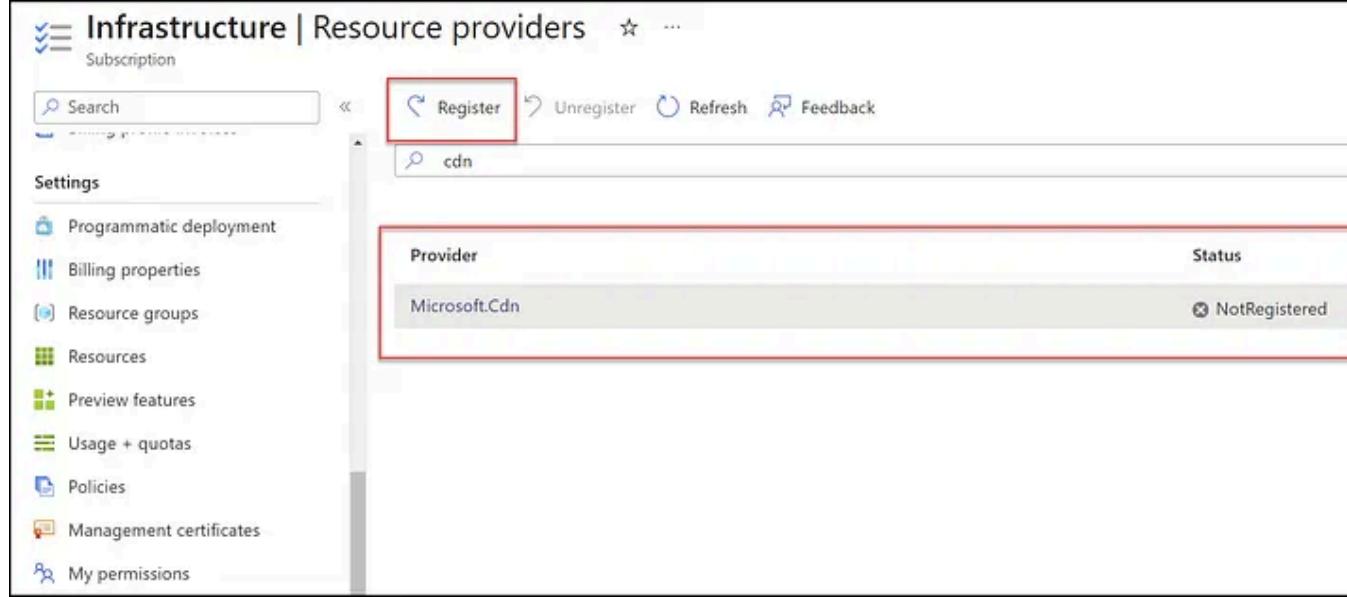


Figure 03 — shows the default Microsoft.Cdn is not registered.

The registration process takes a few seconds to complete. When done, a green check mark will appear next to the Registered Status.

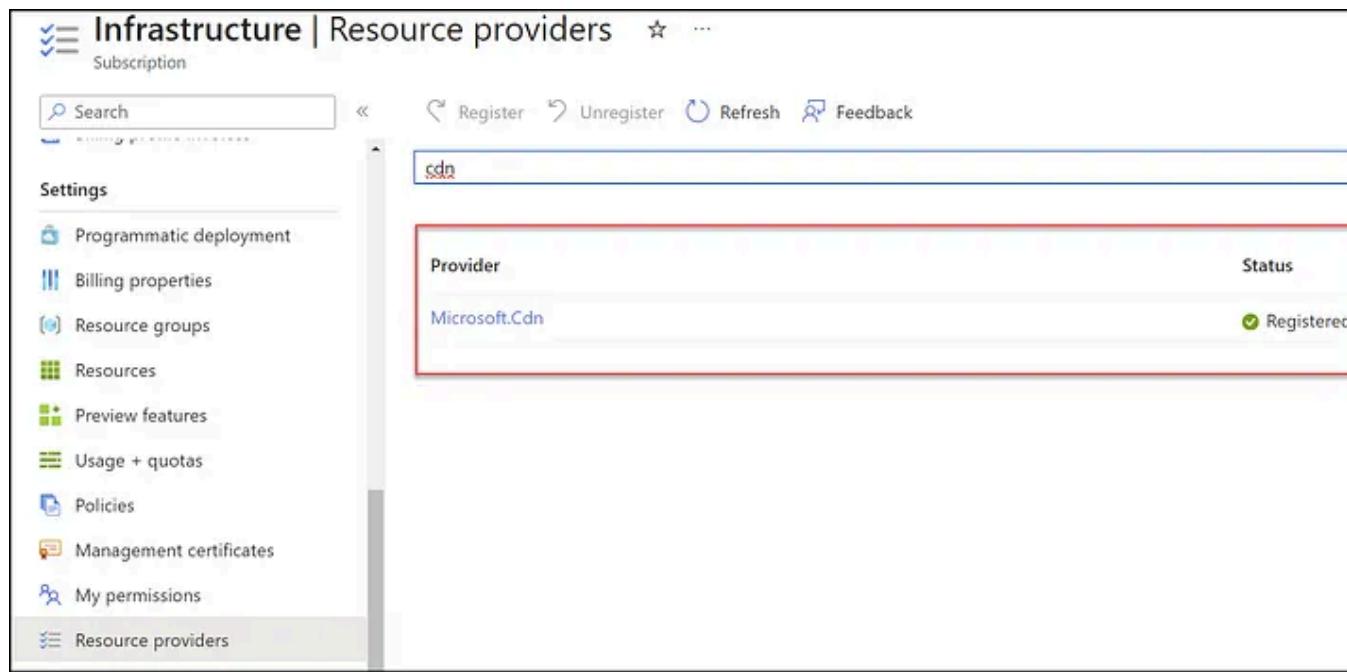


Figure 04 — shows the Microsoft.Cdn status as Registered.

... . . .

Configuring VM Firewall Rules

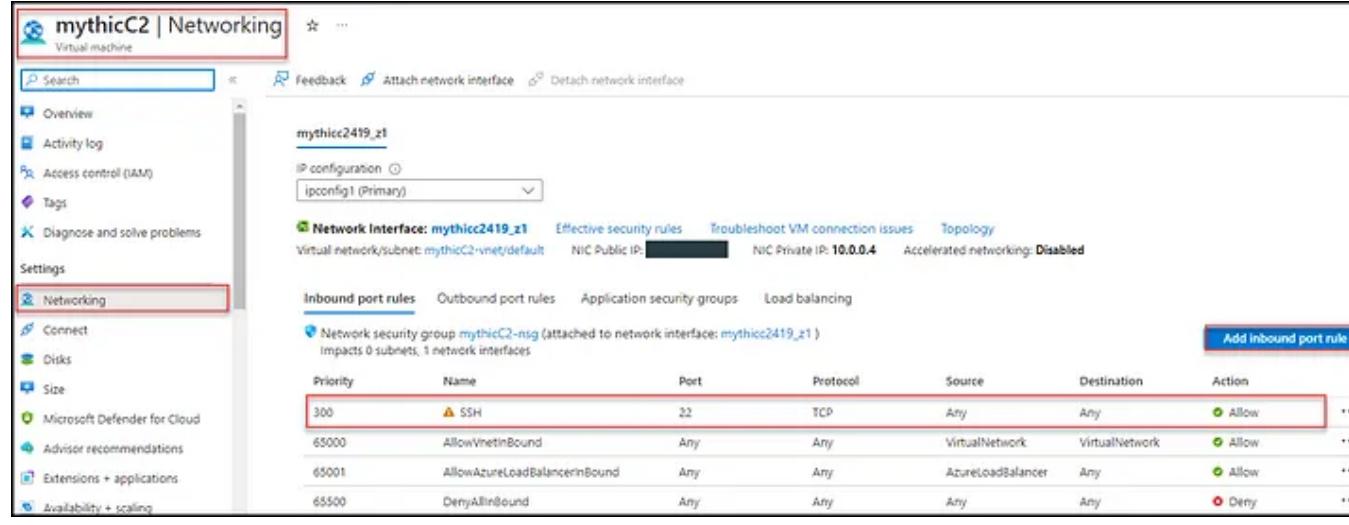
The next step is to set up the firewall rules on the virtual machine hosting the C2 server. For this guide, I'm using an Azure-hosted virtual machine with the Mythic C2 framework deployed on it.

The objective is to open ports 80 and 443 to communicate with the CDN endpoint and the targeted system. If you want to follow along, you can refer to the previous articles:

- [Deploying Azure VMs for C2 Infrastructure](#)
- [Installing Mythic C2 on Azure VM](#)

To add a new firewall rule, we need to select the VM machine and then click on the Networking section. Currently, we have only one (1) rule added,

which is for port 22/TCP for SSH. We will add two (2) more rules; the first one will be for port 80, and the second one will be for port 443.



The screenshot shows the Azure portal interface for a virtual machine named 'mythicC2'. The left sidebar has 'Networking' selected. The main pane displays the 'Network Interface: mythicC2419_z1' settings. Under 'Inbound port rules', there is a table with the following data:

Priority	Name	Port	Protocol	Source	Destination	Action
300	SSH	22	TCP	Any	Any	Allow
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInbound	Any	Any	Any	Any	Deny

A blue button labeled 'Add inbound port rule' is visible at the top right of the table area.

Figure 05— shows the Networking location in the VM settings.

To create a new rule, click on the “Add Inbound port rule” option. In the **Source** field, we can select the specific source IP addresses that we want to allow inbound traffic from. This can be a range of IP addresses in the form of CIDR ranges, or we can use the Microsoft Service tag option, which has predefined identifiers that represent a category of IP addresses, such as the Azure FrontDoor service, which we will be using.

To set up the HTTP port 80 rule, we will follow the below configuration and then click on the “Add” button to add the new rule.

```
** Firewall Rule Configuration for HTTP port 80 **
```

```
Source: Service Tag
Source service tag: AzureFrontDoor.Backend
Source port ranges: *
Destination: Any
Service: HTTP
Destination port range: 80
Protocol: TCP
Action: Allow
Priority: 310
```

Add inbound security rule
mythicC2-nsg

Source ①

Service Tag ←

Source service tag * ①

AzureFrontDoor.Backend ←

Source port ranges * ①

* ←

Destination ①

Any ←

Service ①

HTTP ←

Destination port ranges ①

80 ←

Protocol

Any

TCP ←

UDP

ICMP

Action

Allow ←

Deny

Priority *

310 ←

Name *

AllowTagHTTPInbound ✓

Description

←

Add Cancel Give feedback

Add inbound security rule
mythicC2-nsg

Protocol

Any

TCP ←

UDP

ICMP

Action

Allow ←

Deny

Priority *

310 ←

Name *

AllowTagHTTPInbound ✓

Description

←

Add Cancel Give feedback

Figures 06 & 07 — show the Firewall configs for the HTTP port 80 Inbound rule.

We'll follow the same steps to create the HTTPS rule on port 443.

** Firewall Rule Configuration for HTTPS port 443**

Source: Service Tag
Source service tag: AzureFrontDoor.Backend
Source port ranges: *
Destination: Any
Service: HTTPS
Destination port range: 443
Protocol: TCP
Action: Allow
Priority: 320

Add inbound security rule
mythicC2-nsg

Source ①

Service Tag ←

Source service tag * ①

AzureFrontDoor.Backend ←

Source port ranges * ①

* ←

Destination ①

Any ←

Service ①

HTTPS ←

Destination port ranges ①

443 ←

Protocol

Any

TCP ←

UDP

ICMP

Action

Allow ←

Deny

Priority *

320 ←

Name *

AllowTagHTTPSInbound ✓

Description

←

Add Cancel Give feedback

Add inbound security rule
mythicC2-nsg

Protocol

Any

TCP ←

UDP

ICMP

Action

Allow ←

Deny

Priority *

320 ←

Name *

AllowTagHTTPSInbound ✓

Description

←

Add Cancel Give feedback

Figures 07 & 08 — show the Firewall configs for the HTTPS port 443 Inbound rule.

After completing the setup, we should have three rules added: SSH, HTTP, and HTTPS. The other rules are auto-generated for the load balancer and virtual network when creating the VM.

Inbound port rules	Outbound port rules	Application security groups	Load balancing				
Network security group mythicC2-nsg (attached to network interface: mythiccc2419_z1) Impacts 0 subnets; 1 network interfaces							Add inbound port rule
Priority	Name	Port	Protocol	Source	Destination	Action	...
300	⚠️ SSH	22	TCP	Any	Any	Allow	...
310	AllowTagHTTPInbound	80	TCP	AzureFrontDoor.Bac...	Any	Allow	...
320	AllowTagHTTPSInbound	443	TCP	AzureFrontDoor.Bac...	Any	Allow	...
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow	...
65001	AllowAzureLoadBalance...	Any	Any	AzureLoadBalancer	Any	Allow	...
65500	DenyAllInBound	Any	Any	Any	Any	Deny	...

Figure 09 — shows the Firewall Inbound rules for the Azure VM.

💡 Microsoft publishes their Azure IP ranges and Service Tags on their site in JSON file called “[ServiceTags_Public.json](#)”. This file contains the IPv4 address ranges for Public Azure as a whole, divided by regions and services . This file gets updated weekly.

Azure IP Ranges and Service Tags – Public Cloud	
Important! Selecting a language below will dynamically change the complete page content to that language.	
Select language	English ▾
	Download
Expand all Collapse all	
▼ Details	
Version:	Date Published:
2024.01.08	1/9/2024
File Name:	File Size:
ServiceTags_Public_20240108.json	3.2 MB
This file contains the IP address ranges for Public Azure as a whole, each Azure region within Public, and ranges for several Azure Services (Service Tags) such as Storage, SQL and AzureTrafficManager in Public. This file currently includes only IPv4 address ranges but a schema extension in the near future will enable us to support IPv6 address ranges as well. Service Tags are each expressed as one set of cloud-wide ranges and broken out by region within that cloud. This file is updated weekly. New ranges appearing in the file will not be used in Azure for at least one week. Please download the new json file every week and perform the necessary changes at your site to correctly identify services running in Azure. These service tags can also be used to simplify the Network Security Group rules for your Azure deployments though some service tags might not be available in all clouds and regions. Customers viewing the Effective Security Rules for their network adapters may note the presence of the "special" Azure platform addresses (168.63.129.16, FE80::1234:5678:9ABC:128) which are part of the Azure platform and NOT included in the JSON files. These platform addresses are described in more detail here: https://docs.microsoft.com/en-us/azure/virtual-network/what-is-ip-address-168-63-129-16 . For more information on Service Tags please visit http://aka.ms/servicetags .	

Figure 10 — shows the JSON file for the Azure IP Ranges and Service Tags.

Configuring Azure Front Door CDN Endpoint

There are several CDN options available on Microsoft Azure. These options differ in their capabilities, such as additional security features like Web Application Firewall (WAF), Private Link, Microsoft Threat Intelligence, and Security Analytics. The pricing of each option also varies.

For our objective, we'll use the Azure Front Door offer and select the Quick Create option. To create a new CDN, type CDN in the Azure search bar and select “Front Door and CDN profiles”.

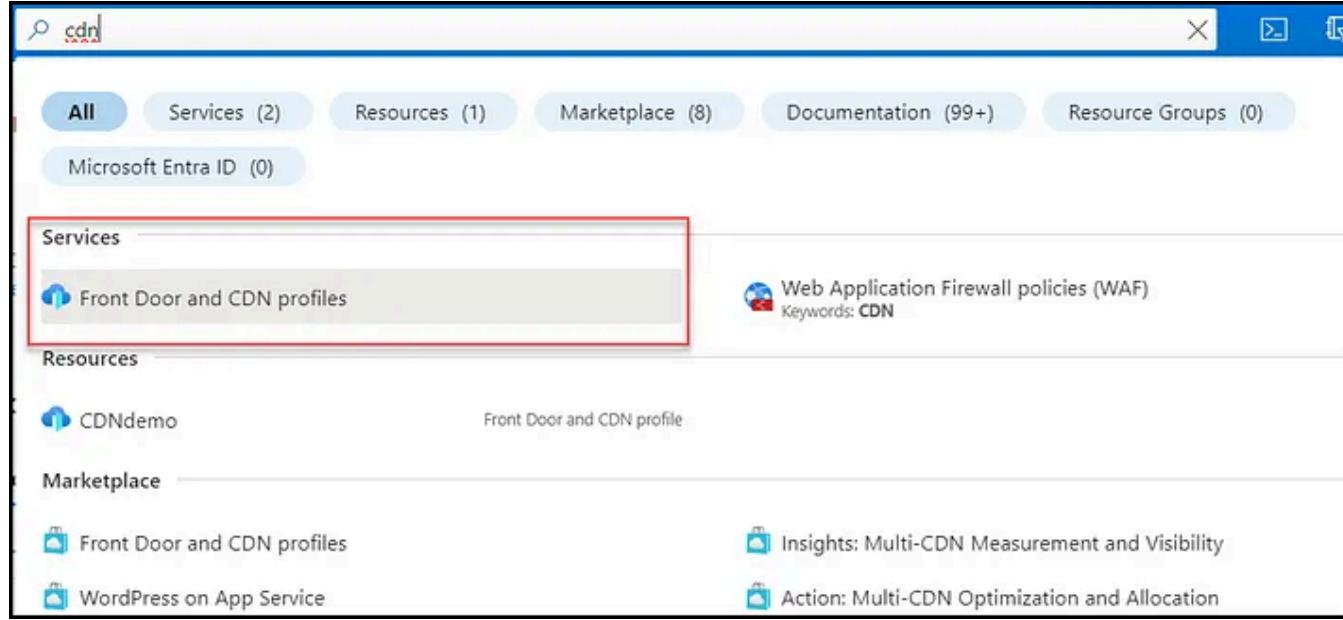


Figure 11 — Shows searching for CDNs in the Azure search bar.

In the CDN dashboard section, we click **Create** to add a new CDN. This dashboard is for listing all of the active CDNs.

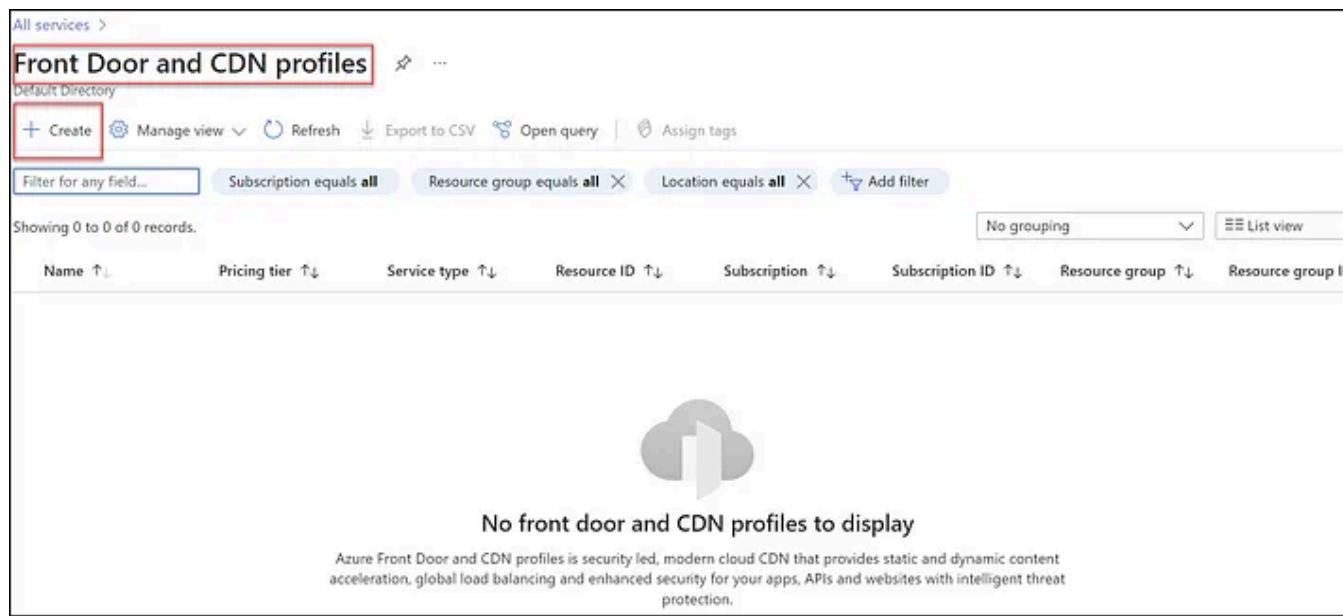


Figure 12 — shows creating a new CDN profile.

Next, click on the **Azure Front Door** and **Quick Create** in the available offering, and click **Continue**.

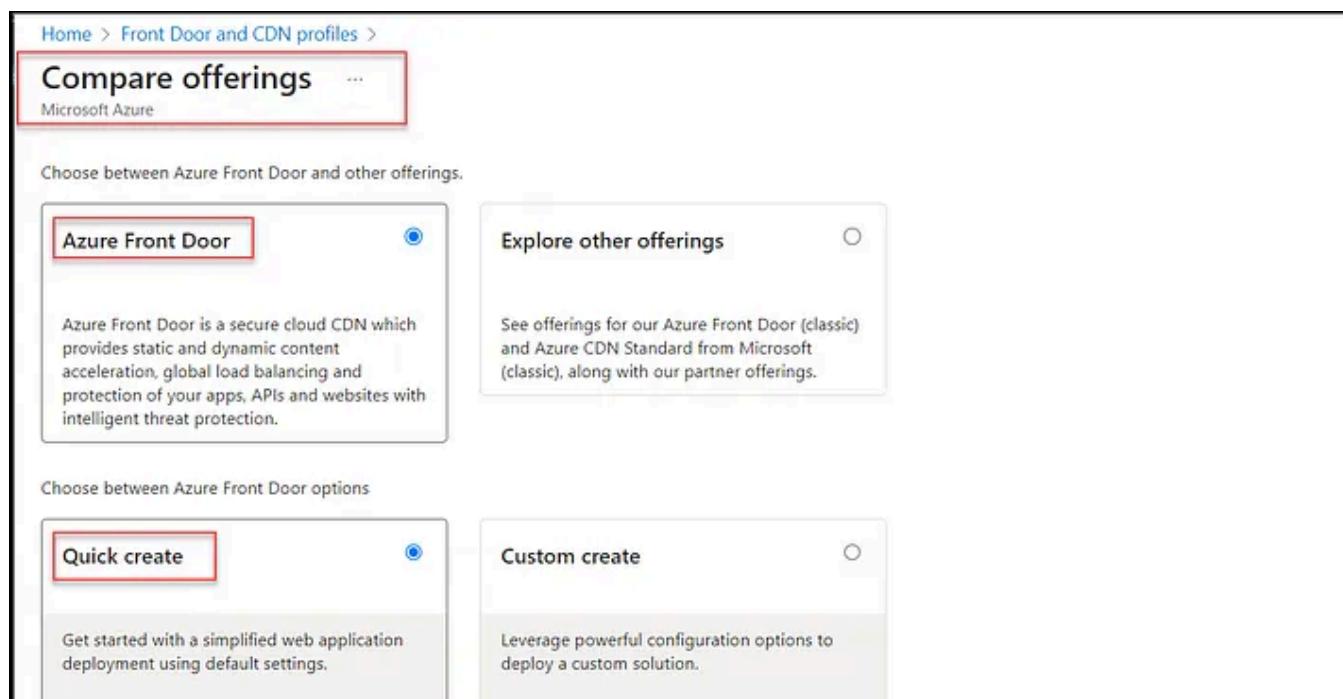


Figure 13 — shows selecting Azure Front Door CDN offer and Quick Create option.

In the Azure Front Door profile, we select the subscription and resource group we previously created, “*infrastructure*” and “*red-ops*”.

After that, name the CDN and select a Tier; we will have it as default at “Standard.” Tier refers to the Azure service capabilities that determine the cost and access latency for retrieving the content. ([Learn more about it on Azure CDN offers](#))

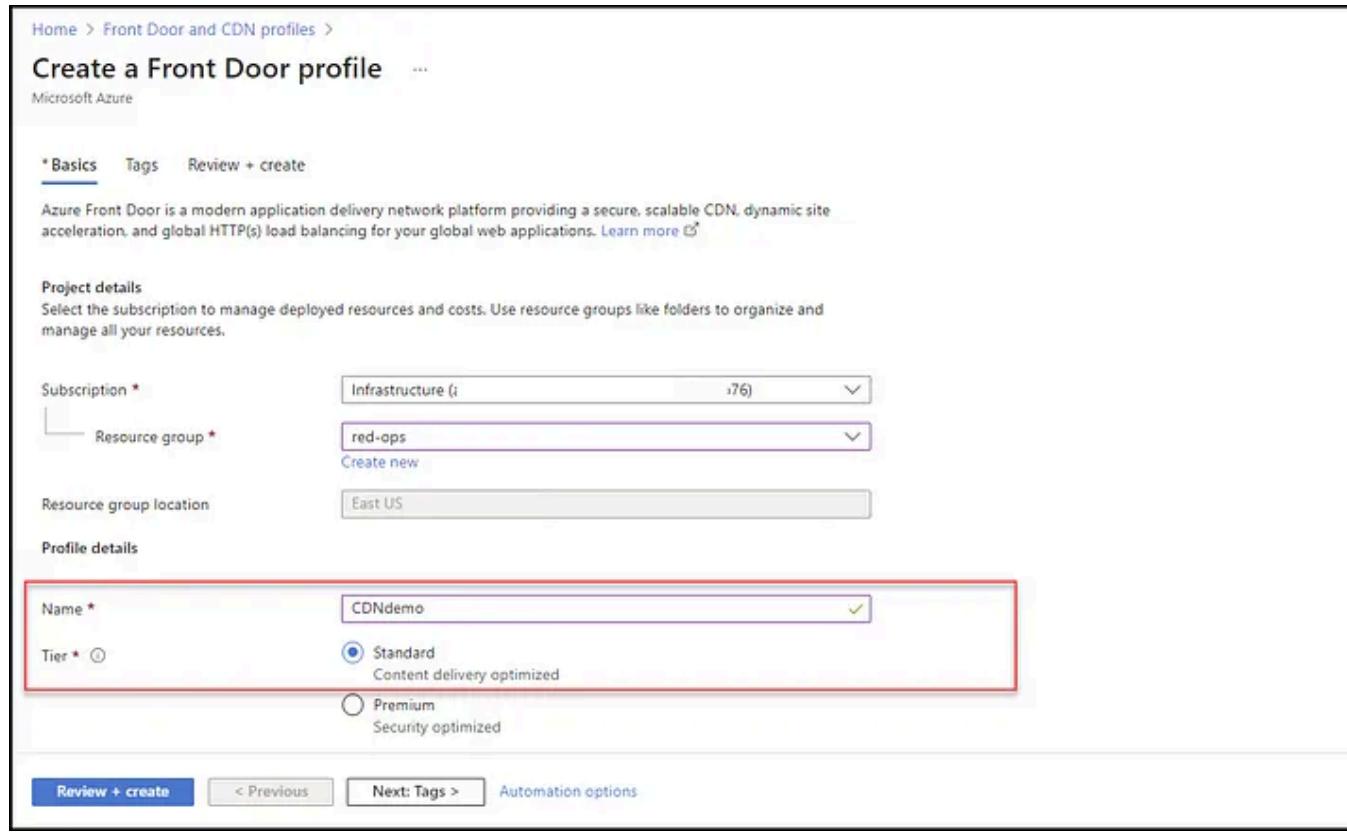
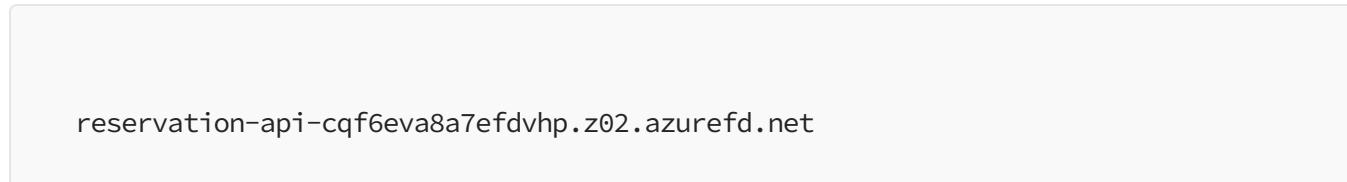


Figure 14 — shows creating the Azure CND Front Door Profile.

To create an endpoint for a CDN, check the box and create a unique name. This endpoint is what the client will see and connect to when the CDN is set up.

It's important to name the CDN something that looks legitimate and blends in with the client's regular expected requests. Avoid sketchy names like “c2 infrastructure” or “redops”. Instead, name it something like “availabilitycalendar” or “reservation-api” if the client is in the hospitality industry or “destinations” or “book-flight” if the client is an airline company.

By default, the URL format for the CDN will follow the `{endpoint-name-randomString}.azurefd.net`, but you can also configure a custom domain name. We'll talk about it in future articles.



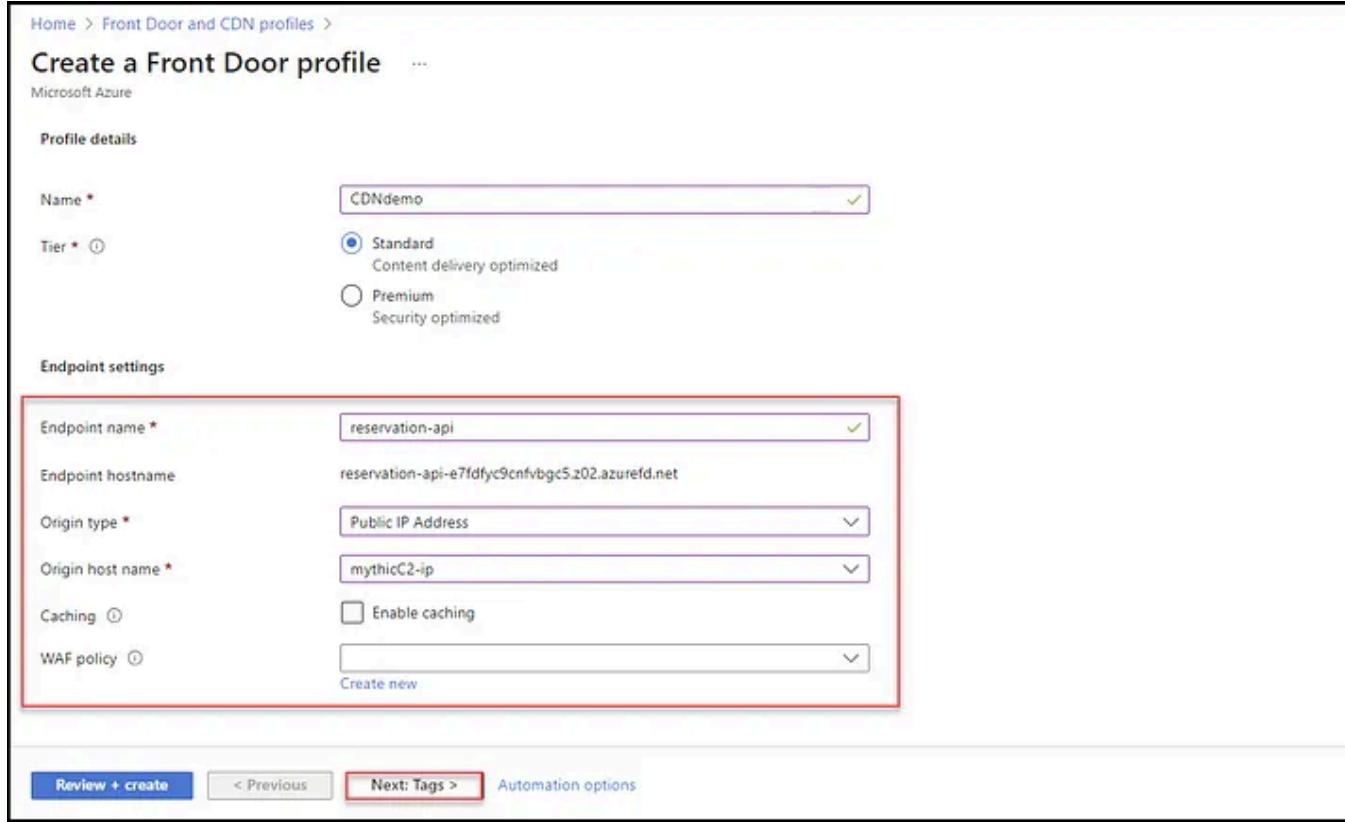


Figure 15 — shows creating the CDN endpoint.

Azure provides various options for **Origin type** depending on the need. The following are the common ones:

- **Storage (Static website)** is for content stored in Azure Storage accounts (blobs, files, containers).
- **Cloud Service** is for Azure Cloud Services.
- **Web App** is for web applications and APIs.
- **Custom origin** is for any other publicly accessible origin web server hosted in Azure or elsewhere.
- **Public IP address**

In this guide, we will refer to the Azure VM that we created in the first part of the guide, "Deploying Azure VMs for C2 Infrastructure," by using its Public IP Address. The hostname, on the other hand, is the name of the VM machine.

To prevent the C2 payload from failing, we ensure that the **Caching** option is unchecked. Enabling caching may cause the payload to be cached at the CDN level if the query string is called twice, which could lead to failure.

For the **WAF policy**, we will keep it as the default — undefined.

Home > Front Door and CDN profiles >

Create a Front Door profile

Microsoft Azure

Profile details

Name * ✓

Tier * Standard Content delivery optimized
 Premium Security optimized

Endpoint settings

Endpoint name * ✓

Endpoint hostname

Origin type *

Origin host name *

Caching Enable caching

WAF policy

Review + create < Previous Next: Tags > Automation options

Figure 16 — shows setting the CND endpoint.

Next, we add **Tags** to our CDN profile and click **Review + Create**. Tags are like labels that can be attached to any Azure service to help you manage and track resource costs.

Basics Tags Review + create

Name ⓘ	Value ⓘ	Resource
name	: red operations	Front Door and CDN profile
	:	Front Door and CDN profile

Review + create < Previous Next: Review + create > Automation options

Figure 17 — shows adding tags to the CDN profile.

Then, review the entered information and click **Create**. Once the CDN is deployed, you'll see a “*Your deployment is complete*” message. Click on “*Go to resource*”.

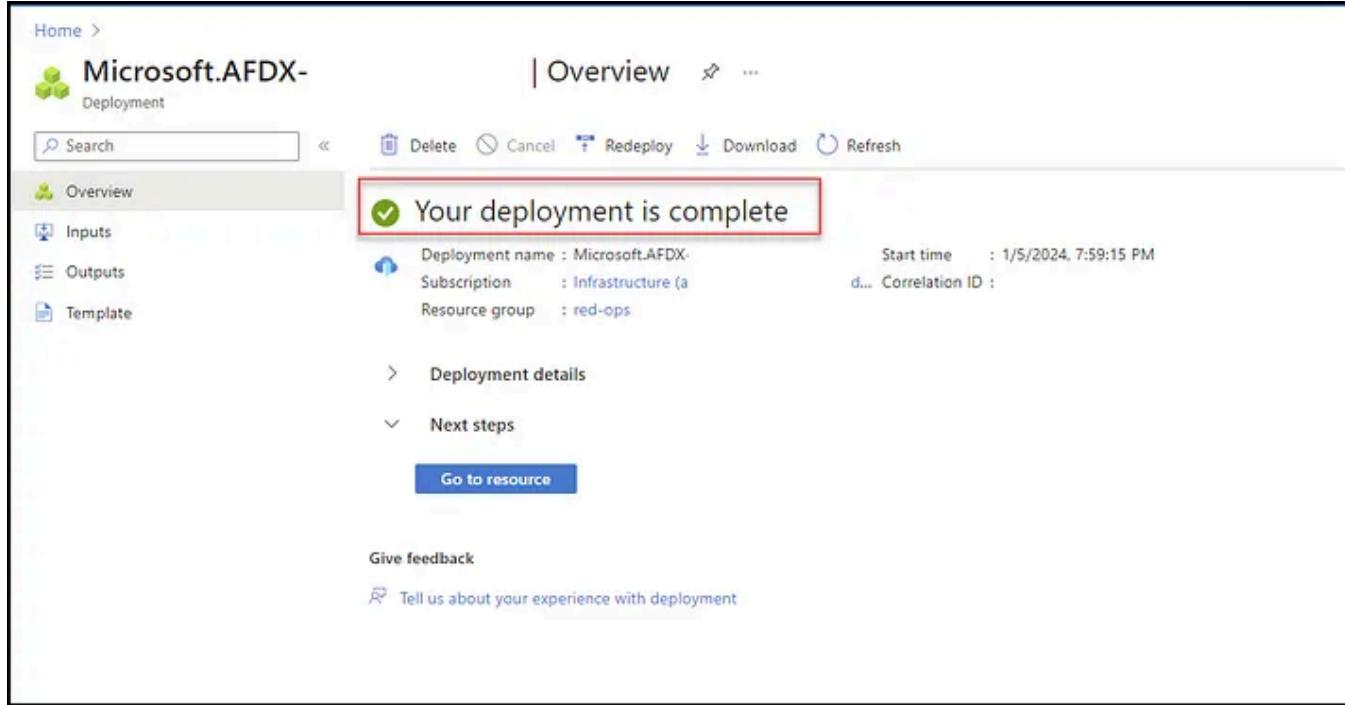


Figure 18 — shows deploying the CND profile.

Accessing the Endpoint

In the newly created CDN profile, the endpoint “*reservation-api*” has been successfully enabled and provisioned. We will make a few changes to the **Routes** and **Origin groups** to meet our objective.

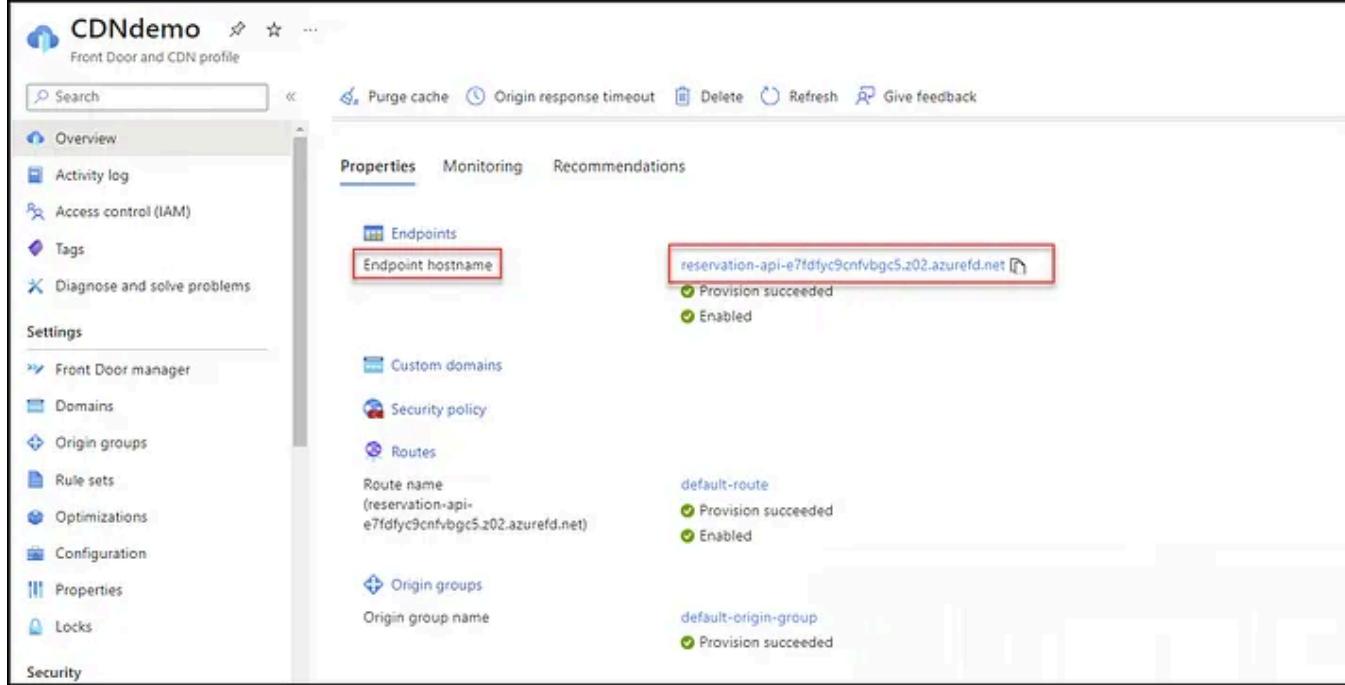


Figure 19 — shows the CDN configurations after deployment.

Routing Options Configuration

In the *Redirector Setup* diagram, we mentioned that client-to-Azure CDN communication takes place over HTTPS on port 443. The reason for this is that Azure CDN redirectors are required to use SSL for encrypted traffic. On the other hand, the CDN communicates with the C2 server via HTTP on port 80.

The screenshot shows the Azure Front Door manager interface. At the top, there are buttons for 'Add an endpoint', 'Refresh', 'Expand all', 'Collapse all', and 'Give feedback'. Below this, a search bar contains the text 'reservation-api-e7fdf...'. There are filters for 'Endpoint enabled status : All' and 'Endpoint provisioning state : All'. A message states: 'You can create multiple endpoints for logical groups of domains. Add routes to connect domains with origin groups. You can add new or use existing domains and origin groups for your routes. Add security rules to complete your Front Door configuration. Learn more'.

The main area displays a table for 'Routes'. It has columns for 'Routes' (with a dropdown arrow), 'Domains' (with a red box around it), 'Origin group' (with a red box around it), 'Status' (Enabled), and 'Provisioning state' (Succeeded). One row is selected, showing 'reservation-api-e7fdf...'. To the right of the table is a 'Security policy' section with a table for 'Name' and 'Domain state', both showing 'No results.'

Figure 20 — shows the default route for the CDN endpoint.

Now that we have created the CDN, clients will communicate with it on port 443. However, to enable the CDN to communicate with the C2 server on port 80, we need to update the route settings of the endpoint.

To do so, click on the “*default route*”, change the forwarding protocol from *HTTPS only* to *HTTP only*, and click **Update**.

The screenshot shows the 'Update route' dialog for the 'default-route'. The 'Endpoint' field is set to 'reservation-api-e7fdf...'. The 'Enable route' checkbox is checked. Under 'Domains', 'Domains' is selected and 'reservation-api-e7fdf...'. Under 'Patterns to match', the value is '/path'. Under 'Accepted protocols', 'HTTP and HTTPS' is selected. In the 'Redirect' section, 'Redirect all traffic to use HTTPS' is checked. Under 'Origin group', 'Origin group' is selected and 'default-origin-group'. Under 'Forwarding protocol', 'HTTP only' is selected (radio button is checked). Under 'Caching', 'Enable caching' is unchecked. At the bottom are 'Update' and 'Cancel' buttons.

Figure 21 — shows changing the default-route forwarding protocol to HTTP only.

Origin groups Configuration

Origins are the servers that receive incoming requests from the Front Door. The *default-origin-group* is a specific set of origins that serves as the primary source for delivering content.

It is the first group of origins that the CDN will attempt to contact when a user requests content. For our redirector, we have the Azure VM in the *default-origin-group*.

The screenshot shows the 'Origin groups' page within the Azure Front Door and CDN profiles. At the top, there are buttons for 'Add', 'Refresh', 'Associate endpoint and route', and 'Give feedback'. A search bar contains the text 'default-origin-group'. Below the search bar, there is a filter 'Provisioning state : All' and a 'Reset' button. The main table has columns: 'Name ↑↓', 'Endpoint ↑↓', 'Routes ↑↓', and 'Provisioning state ↑↓'. One row is visible, labeled 'default-origin-group' with an expanded dropdown menu. Underneath this row, it shows 'reservation-api' as the endpoint, '1 associated route' (labeled 'default-route'), and 'Succeeded' for both provisioning and route states. There is also a three-dot menu icon.

Figure 22 — shows the location of Origin groups inside the CDN profile.

In this section, we will not modify anything except unchecking the *Session Affinity* and *Health Probes* options and clicking the **Update** button.

The screenshot shows the 'Update origin group' dialog for the 'default-origin-group'. The 'Name' field is set to 'default-origin-group'. In the 'Origins' section, there is a table with columns: 'Origin host name', 'Status', 'Priority', and 'Weight'. One entry is shown: 'Azure VM IP' with status 'Enabled', priority '1', and weight '1000'. Below this, under 'Session affinity', there is a checkbox labeled 'Enable session affinity' with a red arrow pointing to it. In the 'Health probes' section, there is a 'Status' field with a red arrow pointing to it, and a checkbox labeled 'Enable health probes' with a red arrow pointing to it. At the bottom, there is a note about load balancing settings.

Figure 23— shows the configuration of the default-origin-group.

The screenshot shows the 'Update origin' dialog box from Microsoft Azure. The form fields are as follows:

- Name:** default-origin
- Origin type:** Custom
- Host name:** (empty)
- Origin host header:** (empty) with a green checkmark icon.
- Certificate subject name validation:** Enable the validation. A note below says: "This validation is required if private link is enabled. Learn more" with a link icon.
- HTTP port:** 80
- HTTPS port:** 443
- Priority:** 1
- Weight:** 1000
- Status:** Enable this origin.

Figure 24— shows the configuration of the origin — Azure VM.

💡 Session affinity is a feature that ensures that requests from a particular user are directed consistently to the same backend server or instance. If this feature is disabled, incoming requests will be randomly directed to any available backend server without maintaining a consistent connection to a specific server throughout the client's session.

💡 Disabling the health probe feature can help reduce additional traffic between the CDN and the C2 server.

CDN Endpoint

After the provisioning process is completed, you might encounter a 405 HTTP response containing an error message stating that “Our services are unavailable right now” while trying to access the endpoint through Curl or a web browser. This happens because rules for HTTP requests need to be established.

However, for our redirector objective, we don’t require it. The payload will connect fine to the CDN endpoint and connect back to us. We’ll see that in the next article when we talk about generating payloads.

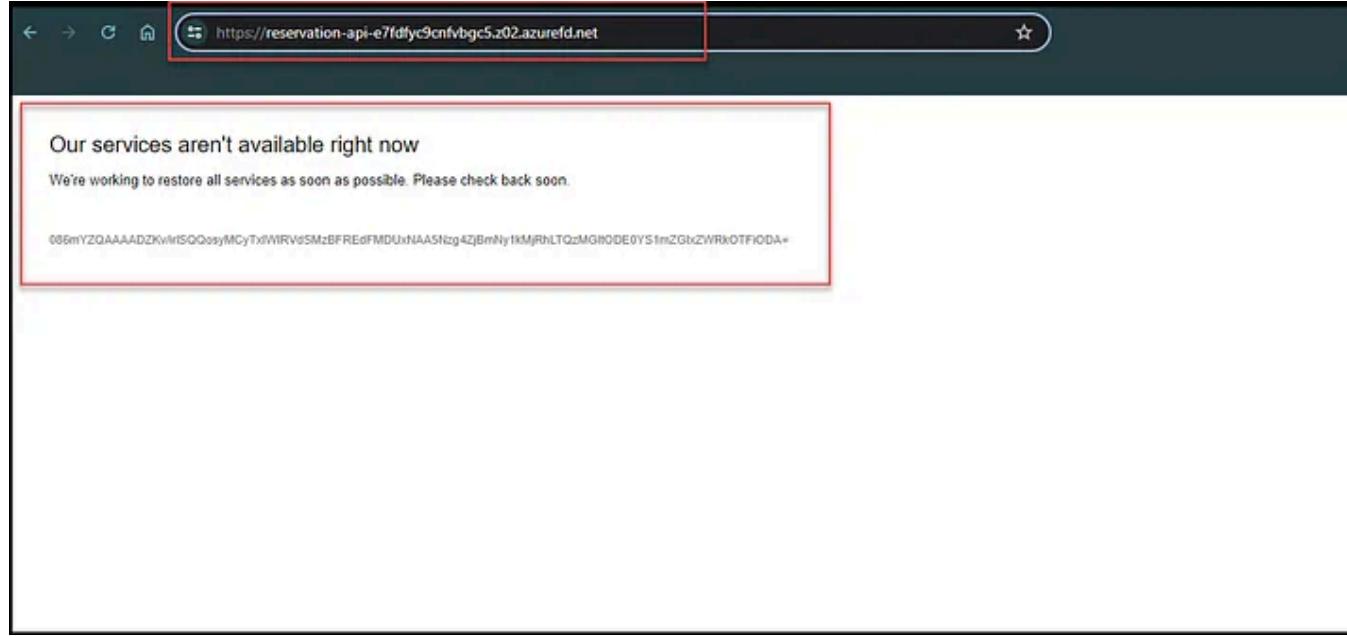


Figure 25 — shows the error when accessing the CDN endpoint in the browser.

```
curl -I https://reservation-api-e7fdfyc9cnfvbgc5.z02.azurefd.net

HTTP/2 405
content-length: 22
content-type: text/plain
x-cache: CONFIG_NOCACHE
x-azure-ref: 0yiagZQAAAADpHDPiaxzxRLySP35acrwwTU5aMjIxMDYwNjEyMDM1ADk30DhmMGY3LWQyNG
date: Thu, 11 Jan 2024 17:35:06 GMT
```

• • •

We have reached the end of today's post. In this article, we have discussed the Azure Front Door CDN service and how it can be configured to serve as a redirector for a red team operation, to obscure the C2 server.

In the next article, we will create a Mythic C2 payload that uses the redirector endpoint we created today to communicate with the compromised machine. To access the previous article, refer to the resources section.

Thank you for taking the time to read this post!

• • •

Resources

- [Microsoft Download Center](#)
- [Azure Front Door CDN Comparison](#)
- [Cyberwarfare Live — Red Team Infra Developer](#)

Red Teaming in the Cloud Series

- [Red Teaming in the Cloud: Installing Mythic C2 on Azure VM](#)
- [Red Teaming in the Cloud: Deploying Azure VMs for C2 Infrastructure](#)

- [Red Teaming in Cloud: Leverage Azure FrontDoor CDN for C2 Redirectors](#)

Hacking

Red Team

Pentesting

Azure

Infosec



20



Published in R3d Buck3T

444 Followers · Last published Feb 7, 2025

Follow

R3d Buck3T focuses on Penetration Testing & Vulnerability Assessment (Red Teaming). My goal is to document what I learn, and share the knowledge with the InfoSec Community



Written by Nairuz Abulhul

1.5K Followers · 21 Following

Follow

[+] Publication: R3d Buck3T

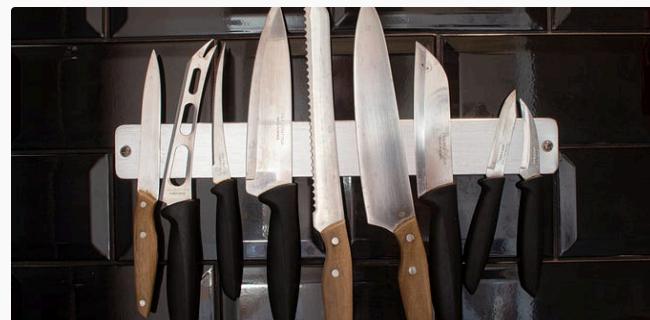
No responses yet



Write a response

What are your thoughts?

More from Nairuz Abulhul and R3d Buck3T



In R3d Buck3T by Nairuz Abulhul

CrackMapExec in Action: Enumerating Windows Networks...

Strategically Mapping Targets inside the Internal Network



In R3d Buck3T by Nairuz Abulhul

Windows PrivEsc with SeBackupPrivilege

Obtain NTLM hashes in Windows Domain Controller machines

Sep 20, 2023 🙌 97 💬 1



Jul 26, 2021 🙌 69



In R3d Buck3T by Nairuz Abulhul

Command Execution with PostgreSQL Copy Command

PostgreSQL access lead to command execution. Supported versions: v9.3-v14

Oct 8, 2021 🙌 17



Mar 3, 2023 🙌 85 💬 2



Cross-Origin Resource Sharing (CORS) Testing Guide

Identifying CORS Vulnerabilities: Common Attack Vectors and Mitigation Strategies

See all from Nairuz Abulhul

See all from R3d Buck3T

Recommended from Medium

Always Free
24 GB RAM + 4 CPU + 200 GB

@harendravarma2 @harendra21 @harendra21

Harendra

How I Am Using a Lifetime 100% Free Server

Get a server with 24 GB RAM + 4 CPU + 200 GB Storage + Always Free

Oct 26, 2024 🙌 9.4K 💬 172



```
(Y) Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "Y"): y
Trying different techniques to give you an edge on the system. This can be invasive
First verifying if you have rights to debug an node to maby exploit the complete node and retrieve some valuable information
Validating permission without token permissions
You have permission to perform create pod
Validating permission without token permissions
You have permission to perform get pods --subresource=exec in
Validating permission without token permissions
Found the node to target
The node Service account token:
Outputting result from cat /host/var/run/secrets/kubernetes.io/serviceaccount/token
SYJ
Fob
LJ
Vid
nD.
u9e
evD
Tic
The ssh authorized keys:
Outputting result from cat /host/root/.ssh/authorized_keys
ian thi
ITxSj
30AvI
57f6h
Retrieving host/travis-kubeconfig:
```

In InfoSec Write-ups by bob van der staak

Penetration testing a Kubernetes environment

Searching for weaknesses in the configuration.

Feb 24 🙌 13



Lists

ChatGPT

21 stories · 991 saves

Natural Language Processing

1981 stories · 1620 saves

Generative AI Recommended Reading

52 stories · 1691 saves

 In Stackademic by Crafting-Code

I Stopped Using Kubernetes. Our DevOps Team Is Happier Than Ever

Why Letting Go of Kubernetes Worked for Us

 Nov 19, 2024  5.9K  173



 Hossam Ehab

PowerShell Exploits—Modern APTs and Their Malicious Scriptin...

Hi in this blog we'll start with an introduction to PowerShell, explaining why it's a favorite...

Feb 14  264  5



 In Offensive Black Hat Hacking ... by Harshad... 

Cybersecurity Roadmap 2025

How to start cybersecurity in 2025?

 Dec 14, 2024  212  4



 Zudonu Osomudeya

100 Essential Commands, Scripts, and Hacks : The DevOps...

100 Essential Commands, Scripts, and Hacks

 Mar 3  172  3



[See more recommendations](#)