Chaos Engineering Scenario: Leverage Chaos Studio to validate the impact of an AAD outage on an application dependency

Introduction

You can use a chaos experiment to verify that your application is resilient to global Azure Active Directory (AAD) outages by causing those outages in a controlled environment. By following this guide, you will simulate an AAD outage using a NSG Security Rule fault and Azure Chaos Studio. The purpose of this simulation is to help you defend and prepare against AAD service outages.

Prerequisites

Before proceeding, you should understand:

- Your service's or application's dependency on Azure Active Directory (https://azure.microsoft.com/services/active-directory/?&ef_id=CjwKCAjwquWVBhBrEiwAt1Kmwp7I8eVJ6-27pOtVtEO9xLok6QmaWqGcqOB9oojv9hU3myL562cMMxoCNJ0QAvD_BwE:G:s&OCID=AID2200277_SEM_CjwKCAjwquWVBhBrEiwAt1Kmwp7I8eVJ6-27pOtVtEO9xLok6QmaWqGcqOB9oojv9hU3myL562cMMxoCNJ0QAvD_BwE:G:s&gclid=CjwKCAjwquWVBhBrEiwAt1Kmwp7I8eVJ6-27pOtVtEO9xLok6QmaWqGcqOB9oojv9hU3myL562cMMxoCNJ0QAvD_BwE)
- Network Security Groups (https://docs.microsoft.com/azure/virtual-network/service-tags-overview)
- AZ Command-Line Interface (https://docs.microsoft.com/cli/azure/)
- The Azure Quality Program (https://eng.ms/docs/quality/program-overview)

Note: For the NSG resource targeted in the experiment, make sure that there is not an NSG outbound rule set to a higher priority that would override the rule to be created during the experiment execution.

Tools

This scenario requires you to have:

- An Azure subscription (https://docs.microsoft.com/azure/guides/developer/azure-developer-guide#understanding-accounts-subscriptions-and-billing). Create a subscription in AIRS (https://azuremsregistration.microsoft.com/Default.aspx) before you begin.
- Access to either Azure CLI or Cloud Shell (https://portal.azure.com/#cloudshell/).

Scenario background

AAD provides single sign-on (SSO), multifactor authentication (MFA), and Conditional Access to guard against most cybersecurity attacks. This helps protect access to resources and data using strong authentication and risk-based adaptive access policies without compromising the user experience. AAD is integral to authenticate numerous applications.

In this scenario, you are temporarily adding a NSG rule using the NSG fault which will cut off communication between AAD and any application running behind that NSG, such as your resource dependencies.

Scenario goal

In this scenario, you will:

- Understand how to evaluate the impact of dependency on AAD and the impact of related outages by using Chaos Studio.
- Identify and use key metrics to formulate an experiment hypothesis.
- Create an experiment that validates the response of a deployed application in the event of an AAD outage.
- Interpret experiment results to assess and potentially reformulate your created hypothesis.

Establish a hypothesis

Establishing a hypothesis is critical before beginning an experiment. Without a hypothesis, it is difficult to understand what to test or how to interpret any results.

For this scenario, create a hypothesis that addresses both AAD and observability expectations. If there is an AAD outage and it causes a failure to resolve one or more of your application's dependencies, what do you expect to happen, and how do you expect to receive the results?

To create a hypothesis, ask questions relevant to the scenario. For example, what resilience measures are already in place to mitigate the impact of an AAD outage? Do these resilience measures work as expected? By running this experiment, what do you expect to happen given your specific application setup? What does a healthy result look like? What is your failure tolerance? What metrics are you assessing?

A hypothesis for this scenario might look like: "In the event of an AAD outage, **ICM incidents were created, and the appropriate resiliency measures were activated. I expect to find the experiment results by analyzing** ." The hypothesis may differ based on your environment.

Using this example, a potential hypothesis may be: "In the event of an AAD outage, no ICM incidents were created, and the appropriate resiliency measures were activated. I expect to find the experiment results by analyzing my application's availability metrics against my defined SLIs and SLOs."

Create an experiment

1. Set up a fault that Chaos Studio will inject into your application or infrastructure using the JSON file below. Save the **JSON file** included below in the same location you are running the **Azure CLI**.

```
{
    "location": "eastus2euap",
    "identity": {
        "type": "SystemAssigned"
    },
    "properties": {
        "steps": [
            {
                "name": "Step1",
                "branches": [
                    {
                        "name": "Branch1",
                        "actions": [
                            {
                                "type": "continuous",
                                "name": "urn:csci:microsoft:networkSecurityGroup:securityRule/1.1",
                                "parameters": [
                                    {
                                        "key": "Name",
                                        "value": "block_aad_fault"
                                    },
                                    {
                                        "key": "Protocol",
                                        "value": "Any"
                                    },
                                    {
                                        "key": "SourceAddresses",
                                        "value": "[\"*\"]"
                                    },
                                    {
                                        "key": "DestinationAddresses",
                                        "value": "[\"AzureActiveDirectory\"]"
                                    },
                                    {
                                        "key": "Action",
                                        "value": "Deny"
                                    },
                                    {
                                        "key": "DestinationPortRanges",
```

```
                                    "value": "[\"*\"]"
                                },
                                {
                                    "key": "SourcePortRanges",
                                    "value": "[\"*\"]"
                                },
                                {
                                    "key": "Priority",
                                    "value": "400"
                                },
                                {
                                    "key": "Direction",
                                    "value": "Outbound"
                                },
                                {
                                    "key": "FlushConnection",
                                    "value": "true"
                                }
                            ],
                            "duration": "PT5M",
                            "selectorid": "Selector1"
                        }
                    ]
                }
            ]
        }
    ],
    "selectors": [
        {
            "id": "Selector1",
            "type": "List",
            "targets": [
                {
                    "type": "ChaosTarget",
                    "id": "/subscriptions/472626f1-3dab-48f5-81e2-d6e1733b586a/resourceGroups/hagurbuz-ses-xms-dev/pro
viders/Microsoft.Network/networkSecurityGroups/test-fc-1/providers/Microsoft.Chaos/targets/microsoft-networksecuritygroup"
                }
            ]
        }
```

```
        ]
    }
}
```

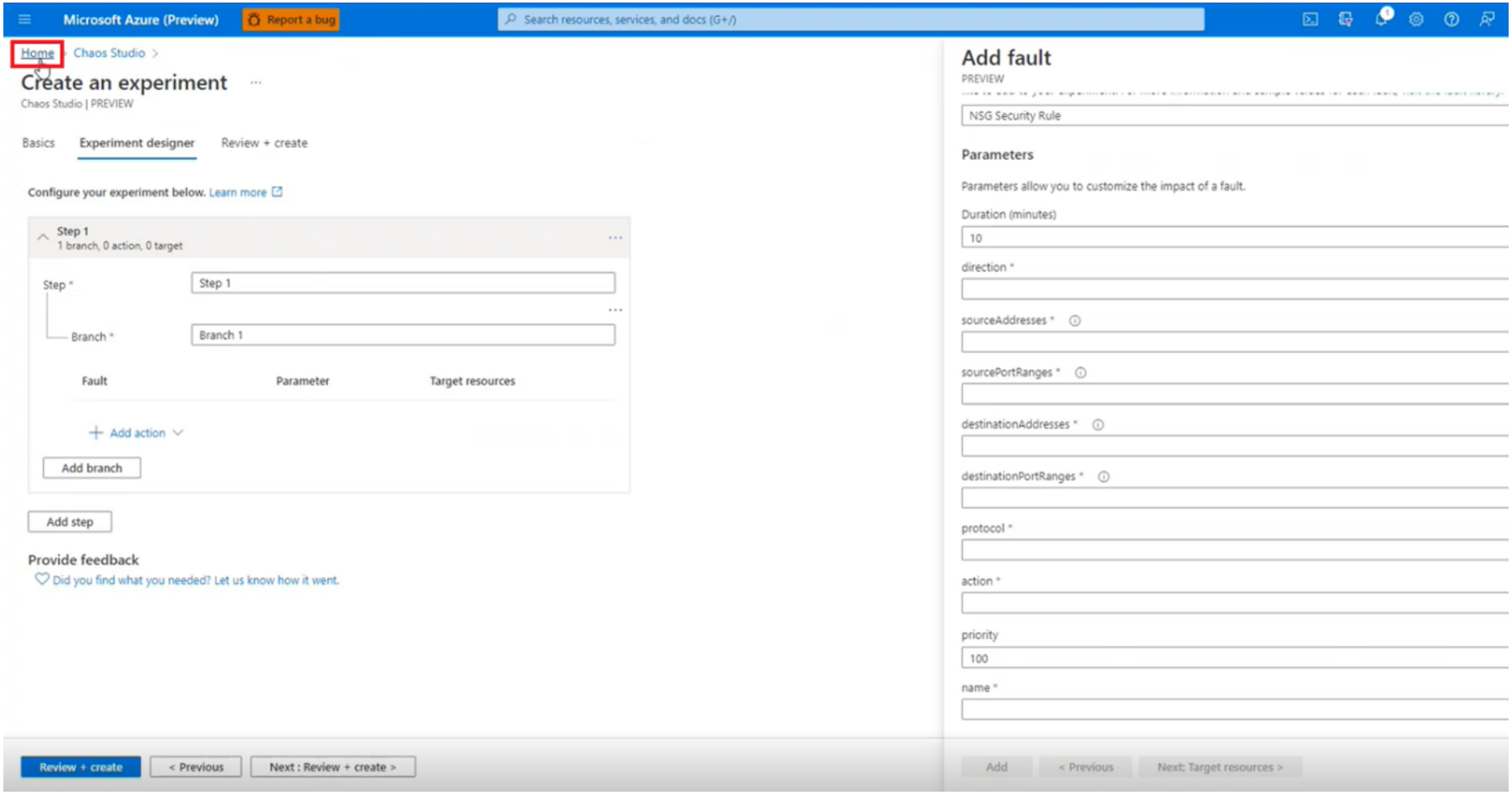
2. Run the following command in your CLI with the **subscription ID**, **resource group**, and **experiment name** replaced to match your experiment.

```
az rest --method put --uri https://management.azure.com/subscriptions/$SUBSCRIPTION_ID/resourceGroups/$RESOURCE_GROUP/prov
iders/Microsoft.Chaos/experiments/$EXPERIMENT_NAME?api-version=2021-09-15-preview --body @experiment.json
```

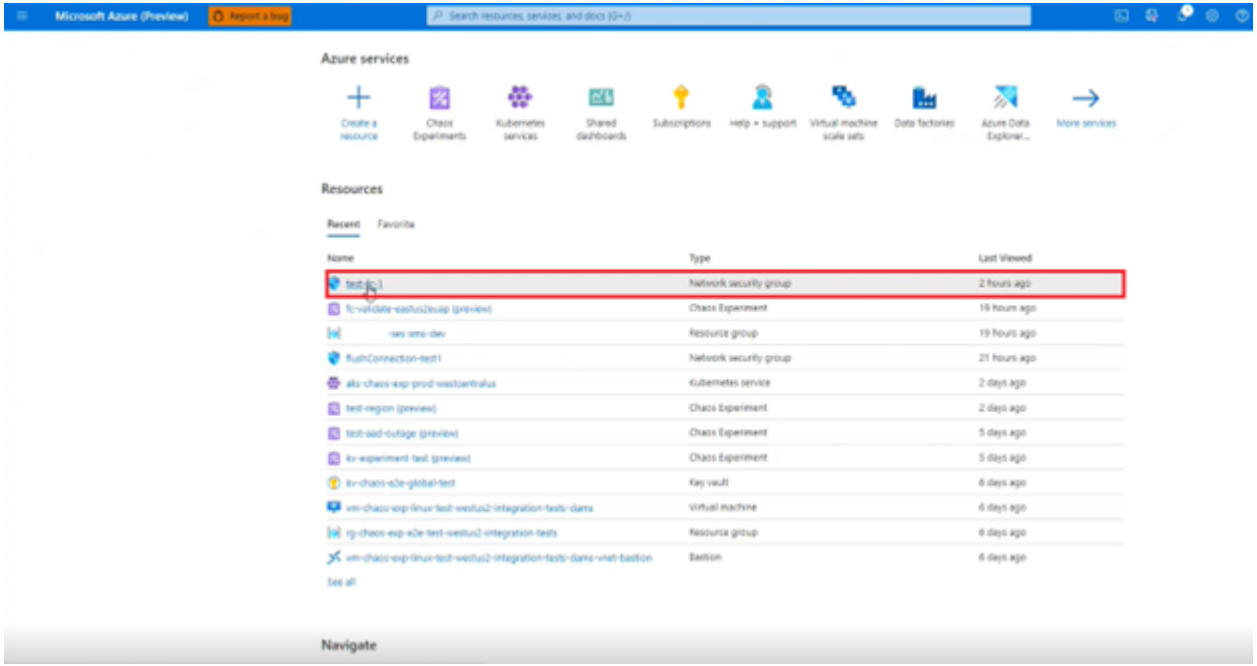
Assign a role to the Network Contributor

Before triggering the experiment, you must assign a role to the Network Contributor. Otherwise, the fault you just created will not run correctly.

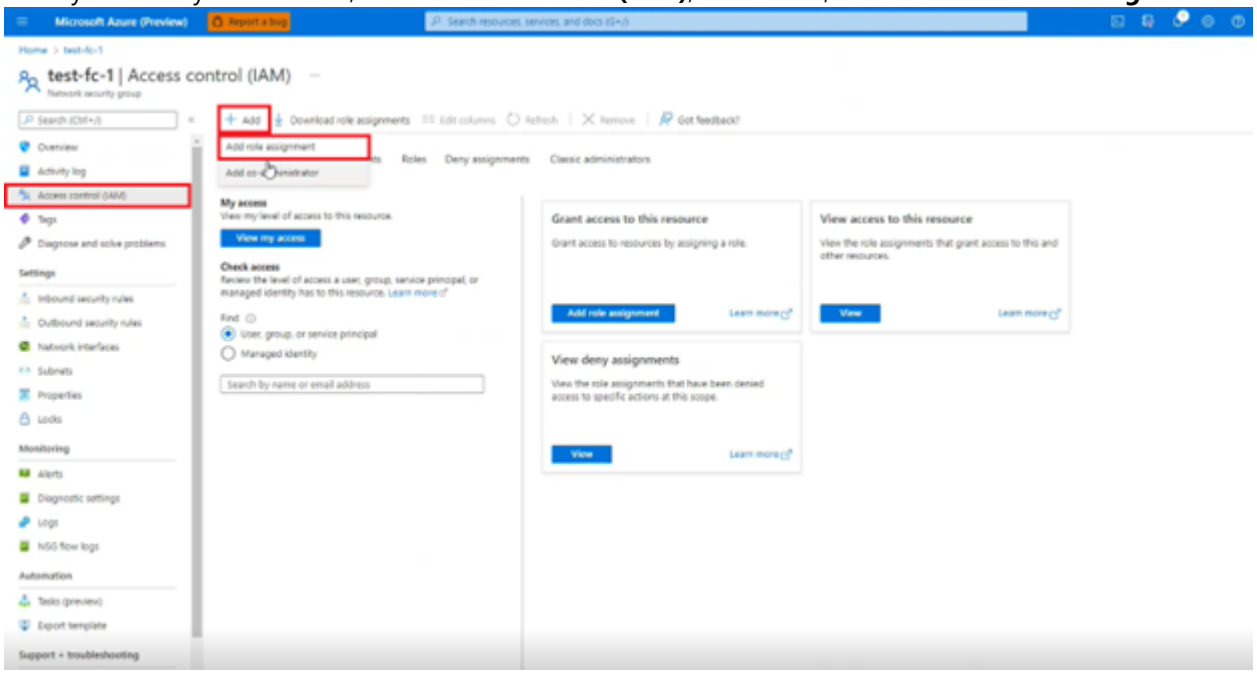
1. In Chaos Studio, select **Home**.



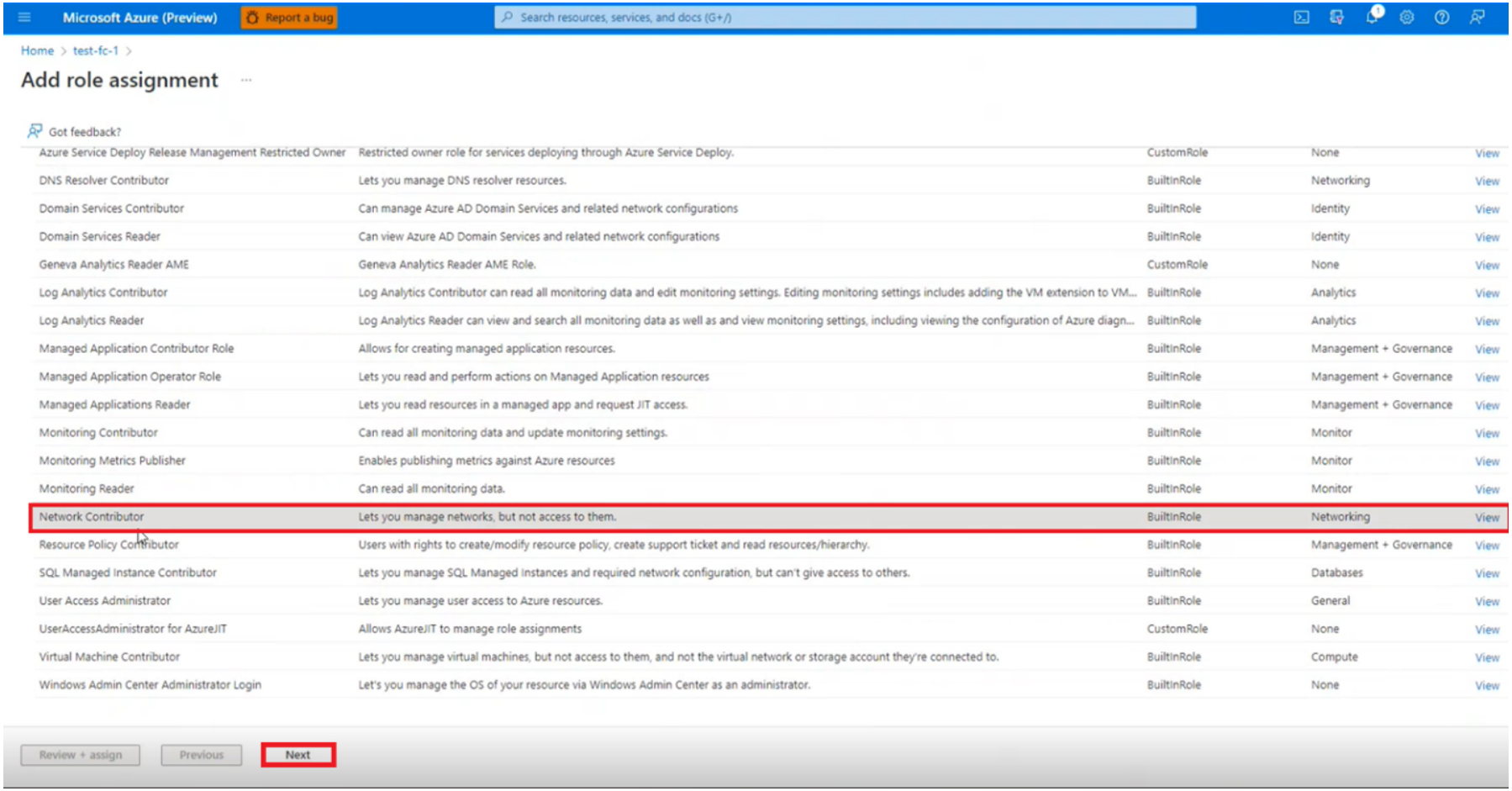
2. Select your resource.



3. Once you select your resource, select **Access control (IAM)**, select **Add**, then select **Add role assignment**.



4. Select **Network Contributor** and then **Next**.



5. Click on **Select members**. Under **Select members**, search for the experiment name, and select **Select** at the bottom of the page. Then, select **Review + assign**.

Microsoft Azure (Preview) Report a bug Search resources, services, and docs (G+/)

Home > test-fc-1 >

Add role assignment

Got feedback?

Role | Members | Review + assign

Selected role Network Contributor

Assign access to
- User, group, or service principal
- Managed identity

Members + Select members

Name	Object ID	Type
No members selected		

Description Optional

Review + assign | Previous | Next

Select members

Select
aad-demo-2

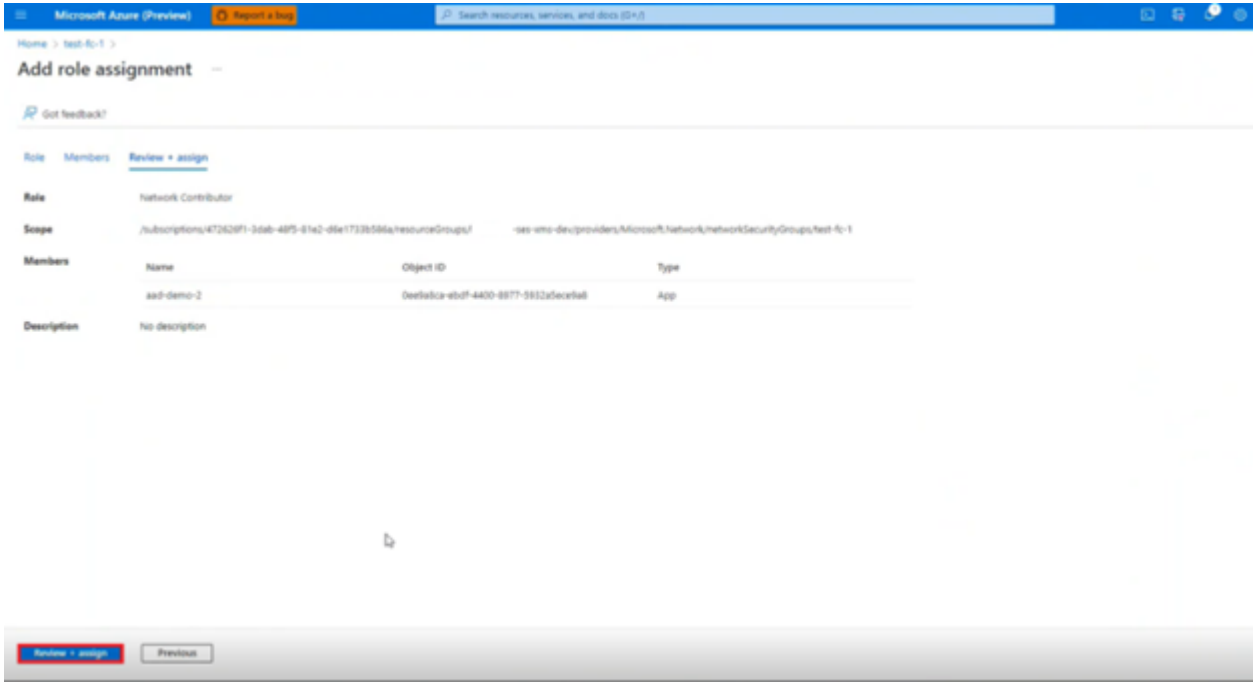
No users, groups, or service principals found.

Selected members:

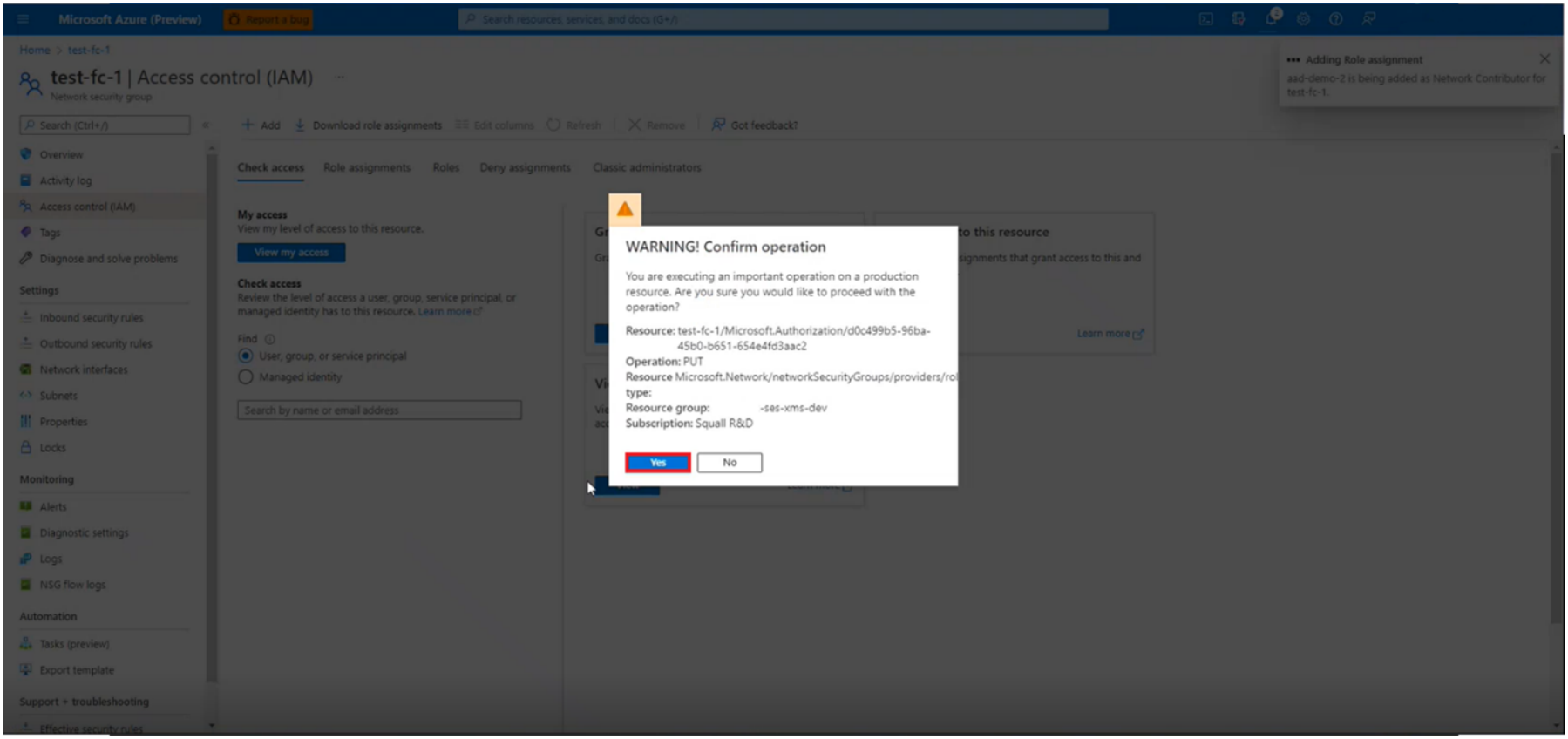
aad-demo-2 Remove

Select | Close

6. Select **Review + assign** again.

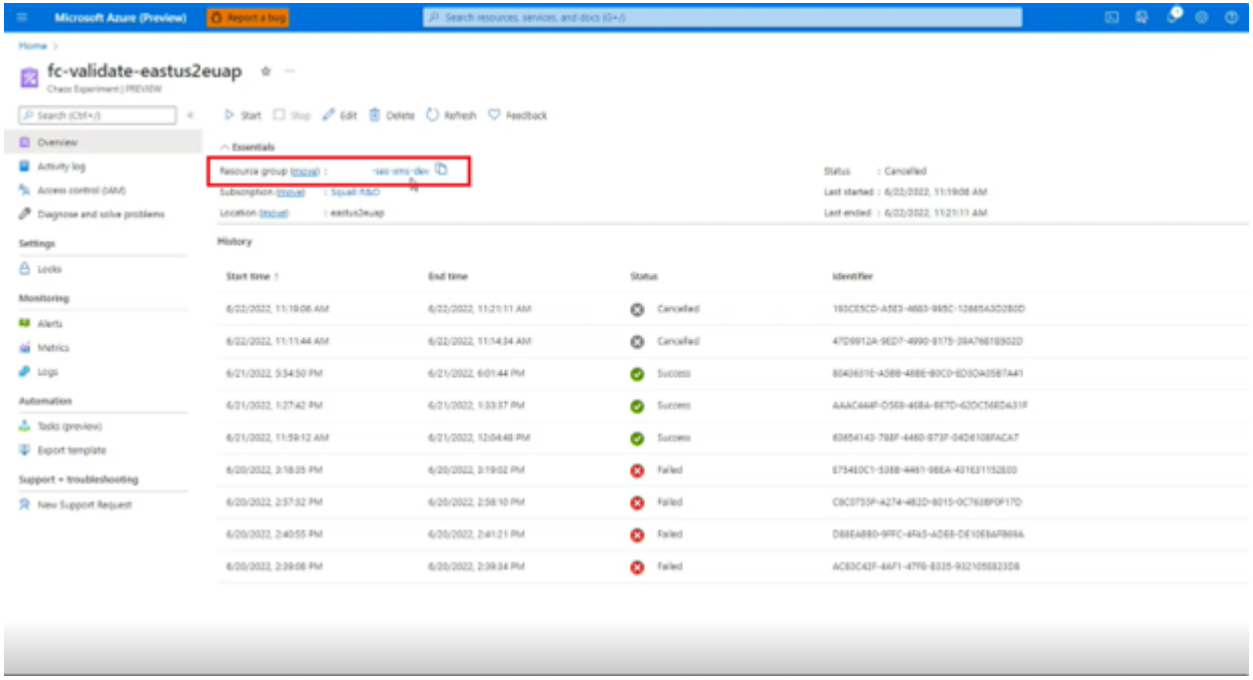


7. Select **Yes** on the **Warning! Confirm operation** box.

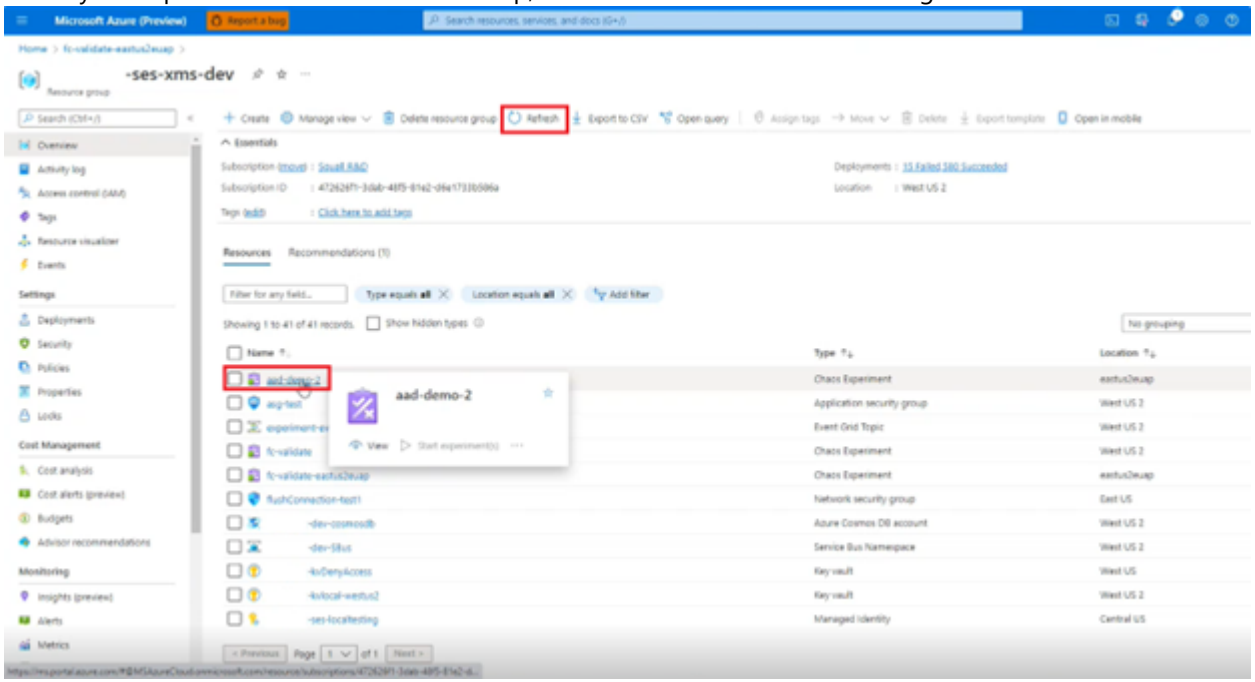


Trigger the experiment

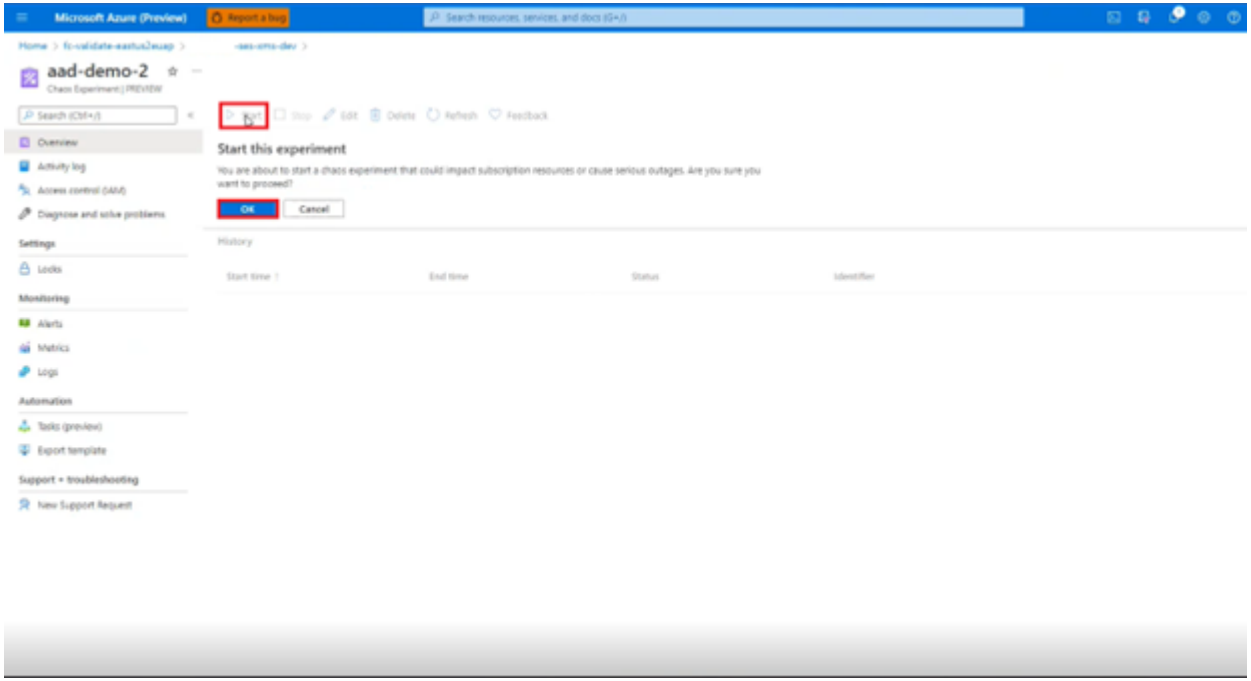
1. Once the **Role assignment** has been successfully added, go back to **Azure Portal** and select your **Resource Group**.



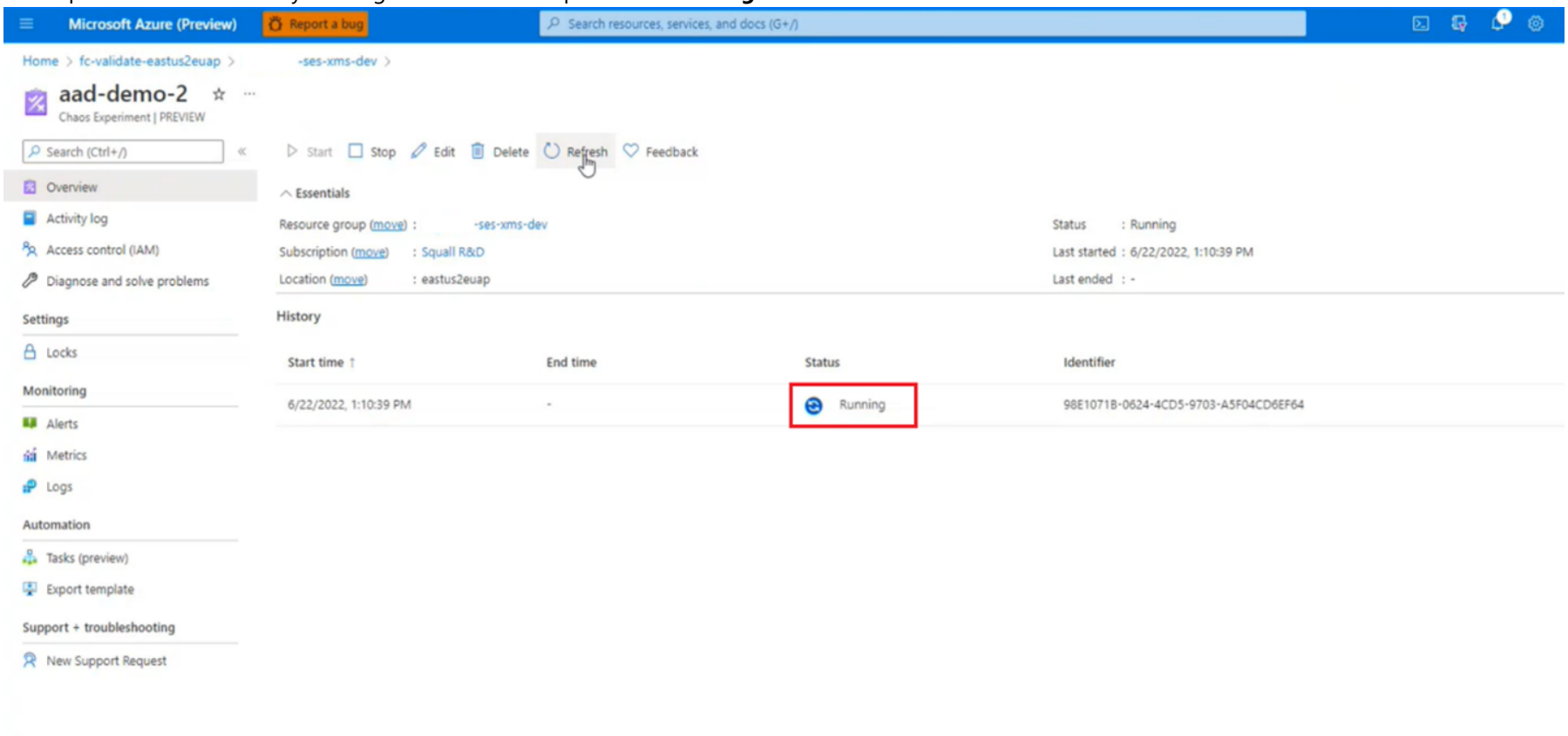
2. Select your experiment. If it does not show up, select **Refresh** and search for it again.



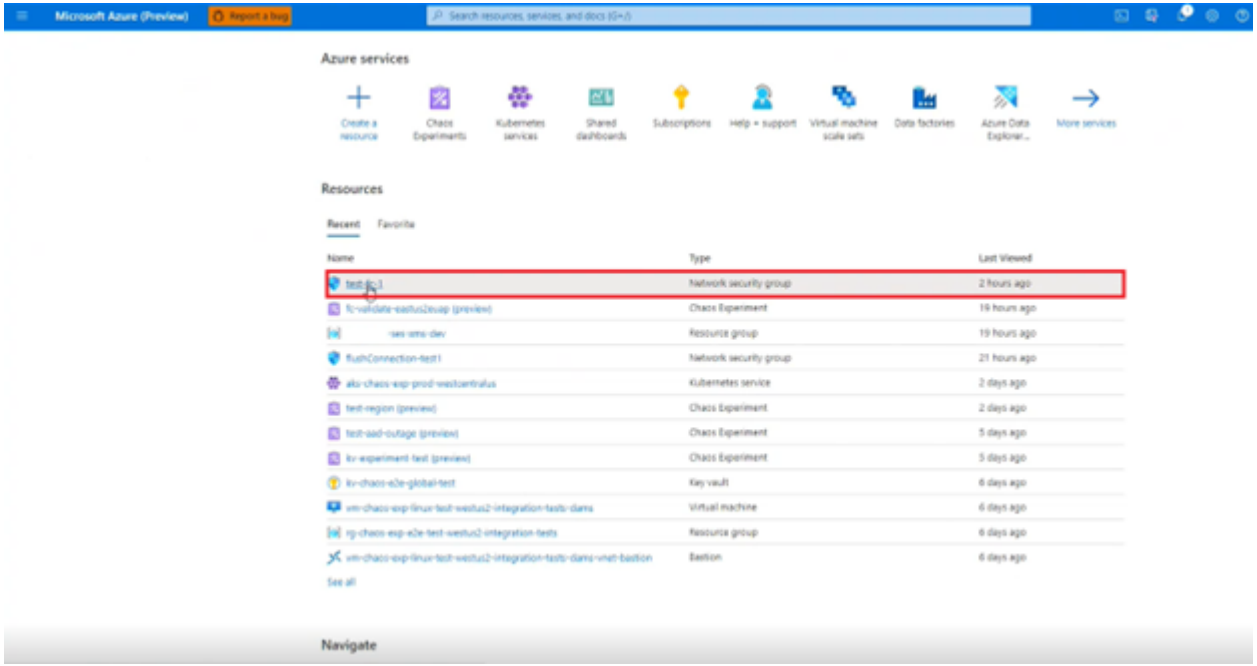
3. Select **Start** and then select **OK**.



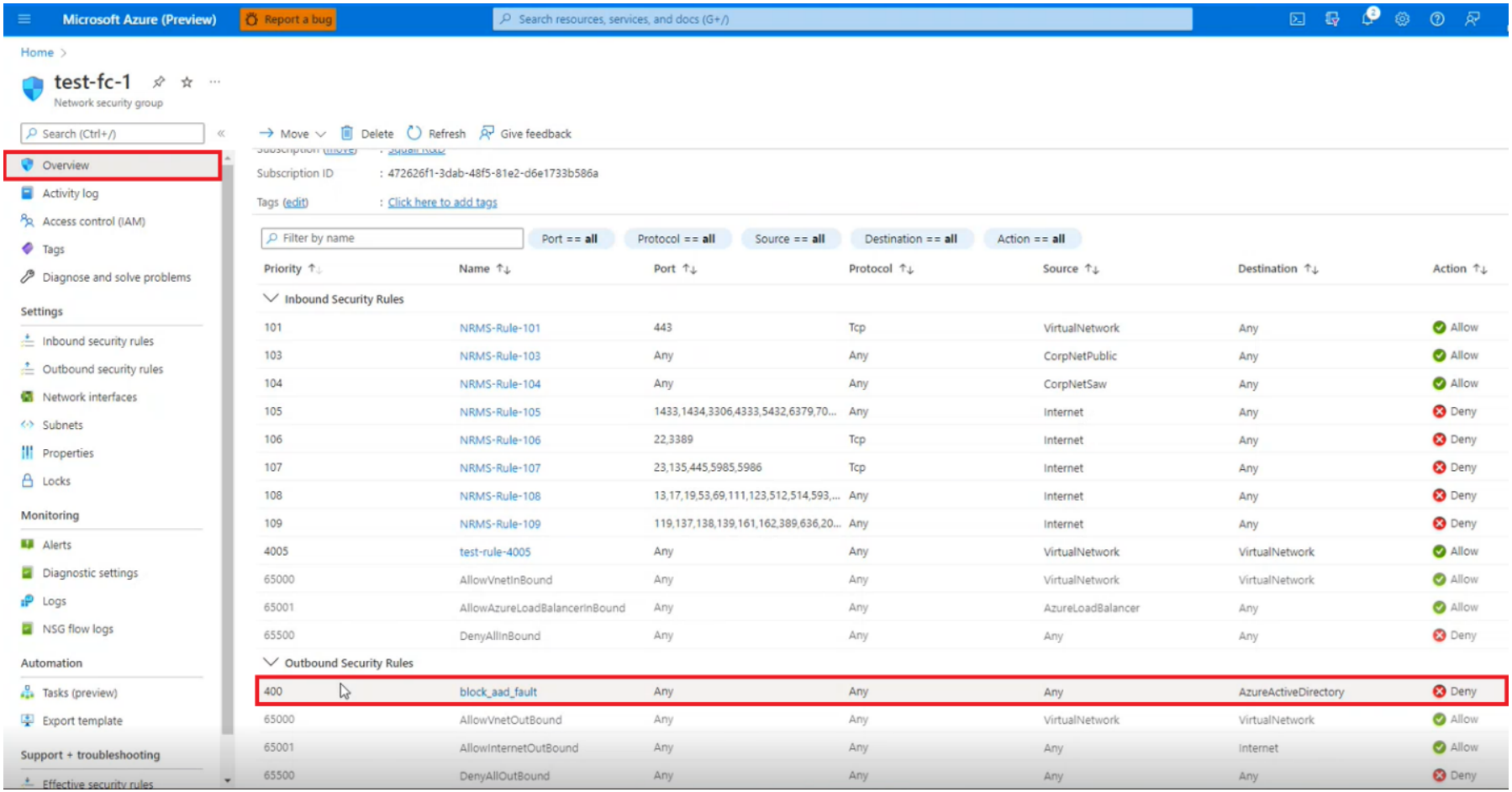
4. The experiment is successfully running once the status updates to **Running**.



5. Once your experiment is running, select **Home** and select your resource.



6. On your resource page, select **Overview** to check for the rule blocking AAD outbound calls. If the rule you inputted says Deny under Action, then it is working correctly and blocking all outbound calls to AAD.



Assess the hypothesis

Compare the results of the experiment against your hypothesis. Analyze any relevant metrics. Do the results align with your expectations?

For example, if your hypothesis addresses availability metrics, analyze your health model in Geneva for the duration of the Chaos experiment to see if there was any impact on the failure rate. If there was an impact, analyze the returned logs and metrics from the experiment to understand why there was a failure rate impact. Similarly, if you are testing to validate SLI alerts or to validate feedback on failures, analyze any feedback against the hypothesis to ensure the alerts are properly responding to failures.

If your results were unexpected, consider any reasons why, create a new hypothesis, implement any necessary changes, and repeat the experiment: “In the event of a AAD outage, **ICM incidents were created, and the appropriate resiliency measures were activated because** resilience improvement has been made. I expect to find the experiment results by analyzing __.”

Overview

You have now learned about AAD and observability metrics, how to formulate and evaluate an experiment hypothesis, and how to create an experiment in Azure Chaos Studio that tests the impact of an AAD outage on an application dependency.

Next steps

- Manage your experiment (https://docs.microsoft.com/azure/chaos-studio/chaos-studio-tutorial-service-direct-portal#:~:text=Manage%20your%20experiment)

Additional resources

- Troubleshoot issues with Azure Chaos Studio (https://docs.microsoft.com/azure/chaos-studio/troubleshooting)
- Chaos Studio fault and action library (https://docs.microsoft.com/azure/chaos-studio/chaos-studio-fault-library)