Software Defined Networking (SDN) in Azure Stack HCI training: Lab for Module 4: Datacenter Firewall

Microsoft Corporation
Published: May 14, 2024

Applies to

SDN training: Module 4: Datacenter Firewall

Copyright

Revision History

Release Date	Changes
May 14, 2024	Initial release.

Contents

SDN LAB: M4 DATACENTER FIREWALL

Overview

Datacenter Firewall is a highly scalable, manageable and diagnosable software-based firewall solution that can be used to secure workloads on a virtual network. Workloads can be moved around to different hosts without breaking configuration and help secure the virtual machines, regardless of guest operating system security implementations implemented by the VM owner.

M4.2 Datacenter firewall architecture

The Datacenter Firewall is composed of the SDN FW service within Network Controller, in addition to the FW Plugin within NCHostAgent. These services are responsible for pushing policies to dataplane to be configured and pushed into VFP where the rules are then applied to traffic traversing the port.

Lab 1: Examine firewall manager service

1. Get the health service state by running:

```
Get-SdnServiceFabricService -ServiceTypeName 'FirewallService'
```

```
PS C:\Users\administrator> Get-SdnServiceFabricService -ServiceTypeName 'FirewallService'


HasPersistedState      : True
ServiceKind            : Stateful
ServiceName            : fabric:/NetworkController/FirewallService
ServiceTypeName        : FirewallService
ServiceManifestVersion : 15.0.82
HealthState            : Ok
ServiceStatus          : Active
IsServiceGroup         : False
```

2. Get the process related to the Firewall Service:

```
Get-Process -Name SDNFW
```

```
[sdn-nc01]: PS C:\Users\Administrator\Documents> Get-Process -Name SDNFW

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)      Id  SI ProcessName
-------  ------    -----      -----     ------      --  -- -----------
   1222      52    82904      87912       9.73    4800   0 SDNFW
```

3. Identify the primary replica for SDN Firewall Service:

```
Get-SdnServiceFabricReplica -ServiceTypeName 'FirewallService' -Primary
```

```
PS C:\Users\administrator> Get-SdnServiceFabricReplica -ServiceTypeName 'FirewallService' -Primary


ReplicaId           : 133132574927249843
ReplicaOrInstanceId : 133132574927249843
PartitionId         : 90db1c75-9205-4d30-8f5b-b25187fc2004
ReplicaRole         : Primary
ServiceKind         : Stateful
Id                  : 133132574927249843
ReplicaStatus       : Ready
HealthState         : Ok
ReplicaAddress      : SDN-NC03.SDN.LAB:0
NodeName            : SDN-NC03
LastInBuildDuration : 00:00:02
```

Lab 2: Examine NCHostAgent

1. Get the svchost process that NCHostAgent is running under.

```
Get-Process -Name svchost | Where-Object {$_.Modules.ModuleName -icontains
"nchostagent.dll"}
```

```
PS C:\Users\administrator> Get-Process -Name svchost | Where-Object {$_.Modules.ModuleName -icontains "nchostagent.dll"}

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)     Id  SI ProcessName
-------  ------    -----      -----     ------     --  -- -----------
    747      49    11692      26196      10.05   2644   0 svchost
```

2. Get the TCP connections used by FW service to connect to OVSDB server. Typically, both vSwitchService and Firewall service will communicate over this port. If no workloads are present on the node, then you may only see a single TCP connection established from NC related to vSwitchService. There will not be a FirewallService connection if no workloads are deployed on the host.

```
Get-NetTCPConnection -OwningProcess 2644 -LocalPort 6640
```

```
PS C:\Users\administrator> Get-NetTCPConnection -OwningProcess 2644 -LocalPort 6640

LocalAddress                        LocalPort RemoteAddress                       RemotePort State       AppliedSetting OwningProcess
------------                        --------- -------------                       ---------- -----       -------------- -------------
::                                  6640      ::                                  0          Listen                     2644
10.184.108.2                        6640      10.184.108.16                       53499      Established Datacenter     2644
10.184.108.2                        6640      10.184.108.14                       58161      Established Datacenter     2644
```

3. Cross-compare the RemoteAddress and RemotePort properties to the primary replicas for vSwitchService and FirewallService to determine the correct TCP thread. For example, in Lab 1, it was identified that SDN-NC03 was the primary replica for the Firewall Service.

```
PS C:\Users\administrator> Enter-PSSession SDN-NC03
[SDN-NC03]: PS C:\Users\Administrator\Documents> Get-NetTCPConnection -LocalPort 53499

LocalAddress                        LocalPort RemoteAddress                       RemotePort State       AppliedSetting OwningProcess
------------                        --------- -------------                       ---------- -----       -------------- -------------
0.0.0.0                             53499     0.0.0.0                           0          Bound                      428
10.184.108.16                       53499     10.184.108.2                      6640       Established Datacenter     428


[SDN-NC03]: PS C:\Users\Administrator\Documents> Get-Process -Id 428

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)      Id  SI ProcessName
-------  ------    -----      -----     ------      --  -- -----------
   1595      66   116432     139460      51.91     428   0 SDNFW
```

M4.3 Network Security Groups

Network Security Groups (previously referred to as Access Control Lists) are the primary element within Network Controller that determine how the Datacenter Firewall is implemented.

Lab 1: Create NSG for subnet

To help ensure consistent security policies for virtual machines attached to a subnet, tenant administrators may want to leverage assigning NSGs to subnets, which ensure that a consistent baseline of firewall rules are applied to all workloads associated with the subnet.

1. Navigate to **WAC** and connect to the cluster **sdnfabric.sdn.lab.**
2. Navigate to **Network Security Groups** > **+New** and specify the name: **Default.**
3. Once created, click on **Default** and under **Rules**, select **+New**.
4. Create the following rules:

Name	Priority	Types	Protocol	Source Address Prefix	Source Port Range	Destination Address Prefix	Destination Port Range	Actions	Logging
TCP_80	101	Inbound	TCP	*	*	*	80	Allow	Enabled
TCP_443	102	Inbound	TCP	*	*	*	443	Allow	Enabled
TCP_3389	103	Inbound	TCP	*	*	*	3389	Deny	Enabled

5. Navigate to **Virtual networks** > **Inventory** > **Contoso-VNET01** > **Subnets (Subnet01)** > **Settings**.
6. Select **Default** under **Network Security Group** and select **Submit**.

Lab 2: Create NSG for NIC

1. Create another NSG, this time with the intent to associate to network interface of a VM directly.
2. Perform the steps 1 through 4 in Lab 1: Create NSG for Subnet, with the following:
    a. Network Security Group:
        i. Name: Allow_RDP
    b. Rules:
        i. Configure TCP_3389 rule, with Actions = Allow (do not configure TCP_80 / TCP_443)
3. Navigate to **Virtual Machines** > **Contoso-VM2** > **Settings** > **Networks** > **Network Security Group** and select **Allow_RDP** and then **Save** your changes.
4. Attempt to perform **ping** or **Test-NetConnection -Port 3389** to the IP address of Contoso-VM1 from Contoso-VM2.
    a. What do you observe? If you recall in Module 3 labs, ping was working between the two VMs once VNET peering was established.

Lab 3: View ACLs (NSGs) within network controller

1. Connect to the SDN-HOST## where Contoso-VM01 is deployed.
2. Return a list of the NSGs configured within the environment.

```
$acls = Get-SdnResource -NcUri "https://ncnorthbound.sdn.lab" -Resource
AccessControlLists
$acls | ConvertTo-Json -Depth 10
```

    a. Locate the ACL that is connected to Contoso-VNET01/subnets/Subnet01.
    b. Locate the ACL that is connected to Contoso-VM1 NIC interface.

Lab 4: View dataplane configuration

SDN FW service will program the underlying ACL into the MS_Firewall database. The FW plugin within NCHostAgent will then calculate the policy and apply it to the VFP API to enforce policy on the port.

1. To confirm the ACL rules have been programmed into the dataplane correctly, navigate to SDN-HOST## where Contoso-VM1 resides.
2. Get the results of the ms_firewall table where InstanceID in the below command is the InstanceID of the rule from previous lab of one of the rules.

```
$ovsdbAcl = Get-SdnOvsdbFirewallRuleTable
$ovsdbAcl | Where-Object {$_.rule_id -ieq "InstanceID"}
```

```
vnic_id           : d1cd3748-7d2e-4c46-bd6e-ca5f773b8ac3
rule_state        : Enabled
action            : Block
dst_ports         : 3389
rule_id           : d0ea146a-e004-41d1-ade3-cc9dc40c35f3
rule_type         : RuleWithControllerPriority
direction         : Inbound
src_ports         : *
priority          : 103
logging_state     : Enabled
src_ip_addresses  : *
uuid              : ea32f73a-c5ea-42f9-86fe-87dce3680dad
protocols         : 6
dst_ip_addresses  : *
```

3. Once the rule has been confirmed to have been added into the ms_firewall table, you can then examine the VFP rules applied to the port. We want to focus on the rules that are inbound in this scenario, since we are attempting to connect from an external endpoint into the VM.

```
$port = Get-SdnVfpVmSwitchPort -VMName Contoso-VM1
Show-SdnVfpPortConfig -PortId $port.PortName -Type IPv4 -Direction IN
```

4. Examine the FW_ADMIN_LAYER_ID and FW_CONTROLLER_LAYER_ID layers.

```
== Layer: FW_ADMIN_LAYER_ID ==
== Group: FW_GROUP_IPv4_IN_ID ==

Rule                                 Priority Flags                    Type  Conditions                                FlagsEx
----                                 -------- -----                    ----  ----------                                -------
9c5dc478-252e-4be5-a12d-8f1ddf0ed0e4      103 8195 terminating stateful allow @{Protocols=6; Destination ports=3389} 0
c5758df9-291d-482d-be4f-fe90f63822e4    65532 8195 terminating stateful block None                                    0
2938bdbb-d810-4175-be68-935a514476e4    65535 8195 terminating stateful allow None                                    0


== Layer: FW_CONTROLLER_LAYER_ID ==
== Group: FW_GROUP_IPv4_IN_ID ==

Rule                                 Priority Flags                    Type  Conditions                                FlagsEx
----                                 -------- -----                    ----  ----------                                -------
d1d65ce4-30d7-4e91-a1f4-39dfb9840f54      101 8195 terminating stateful allow @{Protocols=6; Destination ports=80}   0
69ff395b-7d07-4988-b1b0-d6fb9db61fdc      102 8195 terminating stateful allow @{Protocols=6; Destination ports=443}  0
d0ea146a-e004-41d1-ade3-cc9dc40c35f3      103 8195 terminating stateful block @{Protocols=6; Destination ports=3389} 0
06f55641-d9fb-4bb8-8f00-1b97961def0a    65532 8195 terminating stateful block None                                    0
2938bdbb-d810-4175-be68-935a514476e4    65535 8195 terminating stateful allow None                                    0
```

   a. Which layer maps to the ACLs applied to the Subnet?
   b. Which layers maps to the ACLs applied to the NIC?

5. To examine the VFP rules as .NET object, you can use the following:

```
# get the list of layers applied to the port
Get-SdnVfpPortLayer -PortId $port.PortName

# get the list of groups associated with a particular layer
Get-SdnVfpPortGroup -PortId $port.PortName -Layer 'FW_ADMIN_LAYER_ID'

# get the list of rules applied to a particular group
Get-SdnVfpPortRule -PortId $port.PortName -Layer 'FW_ADMIN_LAYER_ID' -Group
'FW_GROUP_IPv4_IN_ID'
```

Lab 5: Capture packet traces for firewall troubleshooting

In some instances, viewing the configuration may not be sufficient in isolating why traffic is not routing as expected and further diagnostics are required. In these situations when troubleshooting data path issues, network traces on the vmSwitch are good next steps to further isolate what is happening within the VMSwitch and VFP layers.

1. Enable RDP on Contoso-VM1 and Contoso-VM2.

```
reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server" /v
fDenyTSConnections /t REG_DWORD /d 0 /f
netsh advfirewall firewall set rule group="remote desktop" new enable=yes
```

2. From each of the Contoso VMs, run Test-NetConnection -ComputerName <IPAddress> -Port 3389 against the remote VM.

```
while($true){Test-NetConnection -ComputerName 172.16.0.4 -Port 3389}
while($true){Test-NetConnection -ComputerName 172.8.0.4 -Port 3389}
```

   a. What do you notice for behavior between the Contoso VMs that are performing the connection to 3389?

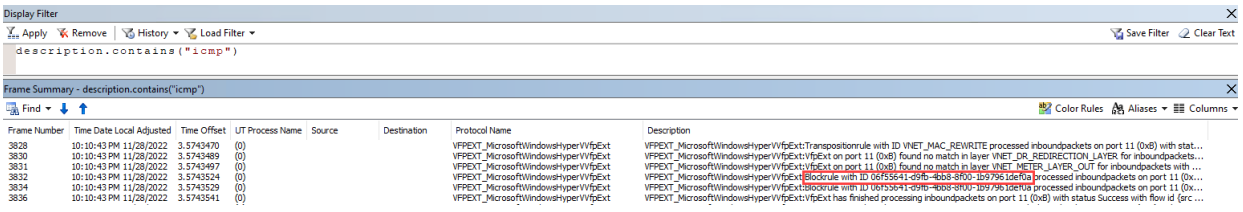
3. On SDN-HOST## where Contoso-VM1 resides performs a packet capture using the following commands, while enabling sufficient time for a repro. Alternatively, you can leverage -ComputerName in the cmdlet if invoking the trace remotely.

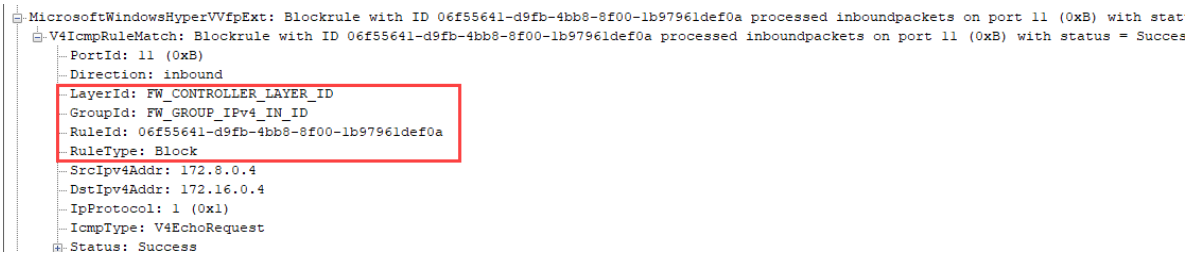
```
Start-SdnNetshTrace -Role Server

Stop-SdnNetshTrace
```

```
[sdn-host02]: PS C:\Users\administrator\Documents> Start-SdnNetshTrace -Role Server
[SDN-HOST02] Required: 2304 MB | Available: 17978.66015625 MB
[SDN-HOST02] Starting netsh trace

Status  FileName
------  --------
Running C:\Windows\Tracing\SdnDataCollection\NetworkTraces\SDN-HOST02_20221128-225054_netshTrace.etl


[sdn-host02]: PS C:\Users\administrator\Documents> Stop-SdnNetshTrace
[SDN-HOST02] Stopping trace

Status
------
Stopped


[sdn-host02]: PS C:\Users\administrator\Documents>
```

4. Copy the _netshTrace.etl file from the host to the lab host and open it with Network Monitor 3.4.

5. Similar to previously done in other labs, you will want to leverage a filter similar to description.contains("string") to search for patterns. In this case looking for ICMP would be sufficient.





6. As you can see, the packet was blocked within VFP due to rule 06f55641-d9fb-4bb8-8f00-1b97961def0a. You can cross-reference the results previously captured in previous lab, query the VFP configuration directly. Update the Layer, Group and Name parameters in the script to match your environment.

```
Get-SdnVfpPortRule -PortId $port.PortName -Layer 'FW_CONTROLLER_LAYER_ID' -
Group 'FW_GROUP_IPv4_IN_ID' -Name 06f55641-d9fb-4bb8-8f00-1b97961def0a
```

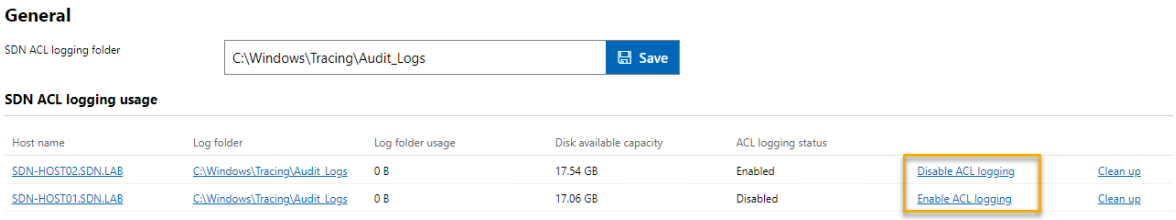
7. Alternatively, you can use Convert-SdnEtwTraceToText on the host where the etl file resides to convert into .txt format, and then be able to see the same type of events as you see in Network Monitor.

```
[0]0000.0000::2022/11/28-22:10:48.752342400 [Microsoft-Windows-Hyper-V-
VfpExt]2(Block)rule with ID 06f55641-d9fb-4bb8-8f00-1b97961def0a processed
1(inbound)packets on port 11 (Name = 9811BF39-D89E-49E6-A00C-BC845608D400,
FriendlyName = NULL) with status = 0x10003(NT=??) and statusLocation = 0:
flow id {src ip = 172.8.0.4, dst ip = 172.16.0.4, protocol = 1, icmp type =
8(V4EchoRequest)}, rule {layer = FW_CONTROLLER_LAYER_ID, group =
FW_GROUP_IPv4_IN_ID, rule id = 06f55641-d9fb-4bb8-8f00-1b97961def0a,
gftFlags = 0}
```

M4.5 Trace logging

Lab 1: Enable flow audit logging

1. Navigate to **WAC** > **sdnfabric.sdn.lab** > **Network Security Groups**.
2. Select **Flow Logs** > **Auditing Settings**.
3. Configure the **SDN NSG Logging Folder** to **C:\Windows\Tracing\Audit_Logs** and then select **Save**.
4. Enable Logging.



5. Confirm you see .json file in the folder on the node(s) tracing was enabled for.
    a. Should show as 0 KB for the current active audit file. Data will be committed to the audit log every hour.