Microsegmentation using NSX Distributed Firewall: Getting Started

VMware NSX for vSphere, release 6.0.x

July 18, 2014

Table of Contents

NOTE: To obtain the latest information about NSX for vSphere, please visit http://www.vmware.com/products/nsx

Introduction

This document guides you through the step-by-step configuration and validation of NSX-v for microsegmentation services. Microsegmentation makes the data center network more secure by isolating each related group of virtual machines onto a distinct logical network segment, allowing the administrator to firewall traffic traveling from one segment of the data center to another (east-west traffic). This limits attackers' ability to move laterally in the data center.

VMware NSX uniquely makes microsegmentation scalable, operationally feasible, and cost-effective. This security service provided to applications is now agnostic to virtual network topology. The security configurations we explain in this document can be used to secure traffic among VMs on different L2 broadcast domains or to secure traffic within a L2 broadcast domain.

Microsegmentation is powered by the Distributed Firewall (DFW) component of NSX. DFW operates at the ESXi hypervisor kernel layer and processes packets at near line-rate speed. Each VM has its own firewall rules and context. Workload mobility (vMotion) is fully supported with DFW, and active connections remain intact during the move.

This paper will guide you through two microsegmentation use cases and highlight steps to implement them in your own environment.

Use Case and Solution Scenarios

This document presents two solution scenarios that use east-west firewalling to handle the use case of securing network traffic inside the data center. The solution scenarios are:

- **Scenario 1**: Microsegmentation for a three-tier application using *three different layer-2 logical segments* (here implemented using NSX logical switches connected over VXLAN tunnels):

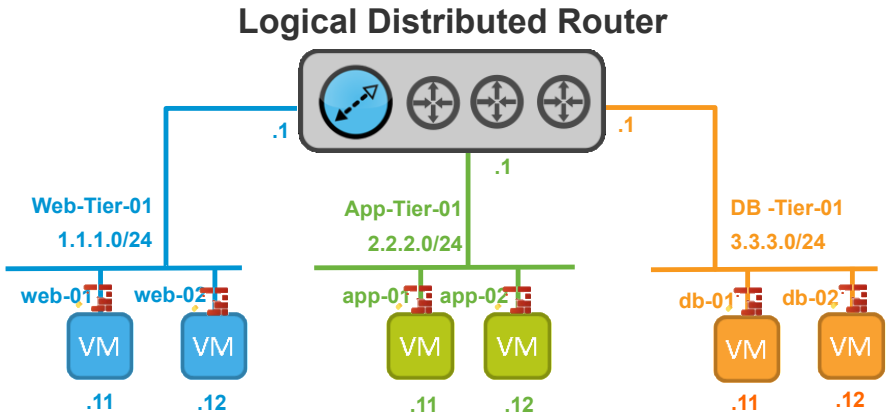


Figure 1 – Scenario 1 Logical View.

In *Scenario 1*, there are two VMs per tier, and each tier hosts a dedicated function (WEB / APP / DB services). Traffic protection is provided within the tier and between tiers. Logical switches are used to group VMs of same function together.

- **Scenario 2**: Microsegmentation for a three-tier application using *a single layer-2 logical segment*:

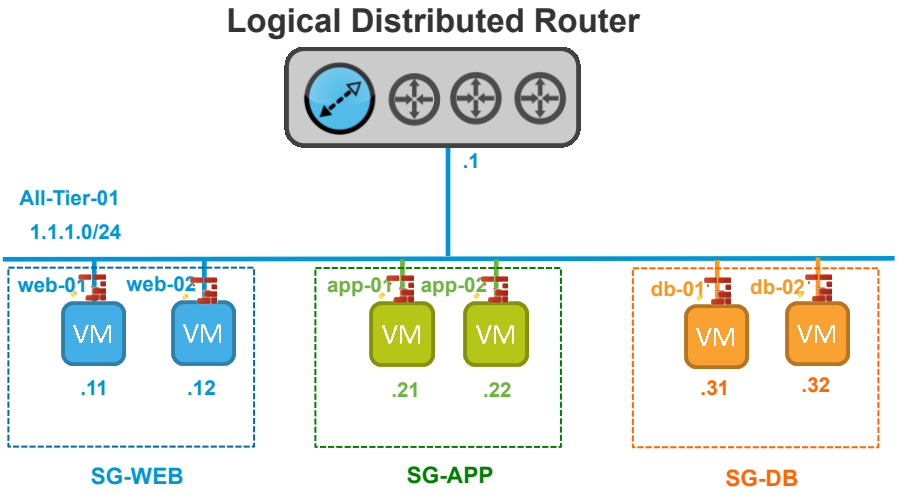


Figure 2 – Scenario 2 Logical View.

In *Scenario 2*, all VMs are located on same tier. Traffic protection is provided within tier and per function (WEB/ APP/ DB services). Security Groups (SG) are used to logically group VMs of same function together.

For both Scenario 1 and Scenario 2, the following security policies are enforced:

Source	Destination	Service	Action	Description
Any Web VM	Any Web VM	HTTP (port 8080)	Allow	Only allow HTTP traffic within Web tier
Any App VM	Any App VM	ICMP	Allow	Only allow ICMP traffic within App tier
Any DB VM	Any DB VM	ICMP	Allow	Only allow ICMP traffic within DB tier
Any Web VM	Any App VM	SSH	Allow	Only allow SSH from Web tier to App tier
Any App VM	Any DB VM	TCP port 1433	Allow	Only allow TCP port 1433 from App tier to DB tier
Any	Any	Any	Block	Default rule set to block action

For *Scenario 1*, a logical switch object is used for source and destination fields. For *Scenario 2*, a Service Composer / Security Group object is used for source and destination fields. By using these vCenter-defined objects, we optimize the number of needed firewall rules irrespective of number of VMs per tier (or per function).

NOTE: TCP port 1433 simulates the SQL service.

NOTE: This section assumes the *Scenario 1* set-up is performed before the *Scenario 2* set-up. In other words, the order of operations starts with Scenario 1 followed by Scenario 2 (going the other way is not handled in this document).

Physical Topology

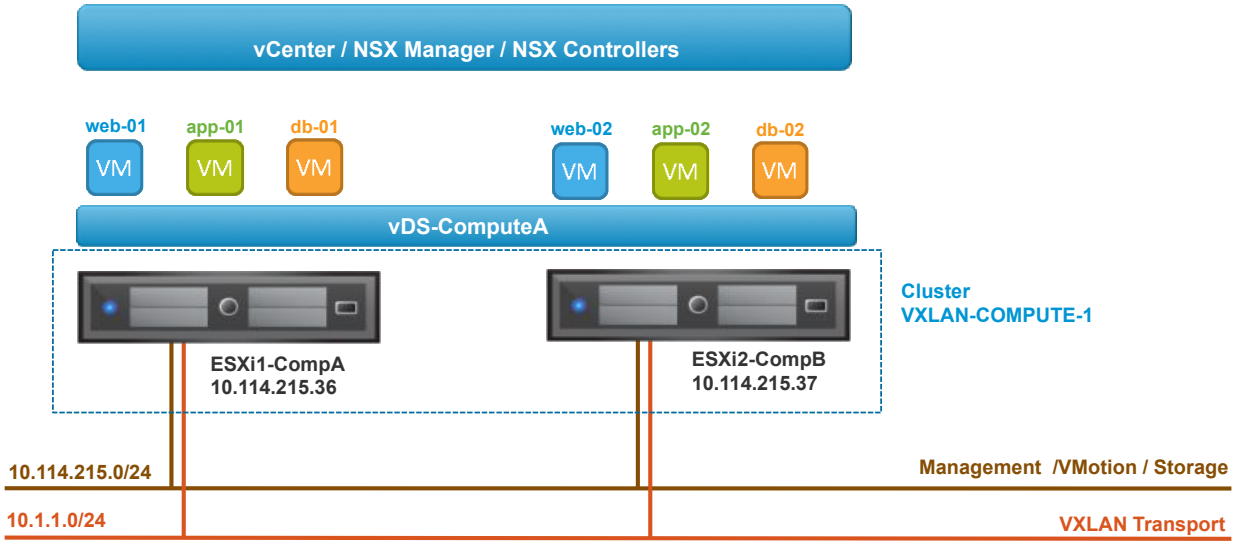


Figure 3 – Physical Topology.

Two ESXi hosts in the same cluster are used. Each host has following connectivity to the physical network:

- one VLAN for management, vMotion, and storage. Communication between the ESXi host and the NSX Controllers also travels over this VLAN.
- one VLAN for data traffic: VXLAN-tunneled, VM-to-VM data traffic uses this VLAN.

Locations:

- Web-01, app-01 and db-01 VMs are hosted on the first ESXi host.
- Web-02, app-02 and db-02 VMs are hosted on the second ESXi host.

The purpose of this implementation is to demonstrate complete decoupling of the physical infrastructure from the logical functions such as logical network segments, logical distributed routing and DFW.

In other words, microsegmentation is a logical service offered to an application infrastructure irrespective of physical component. There is no dependency on where each VM is physically located.

Looking at the vCenter web interface, the topology looks like this:

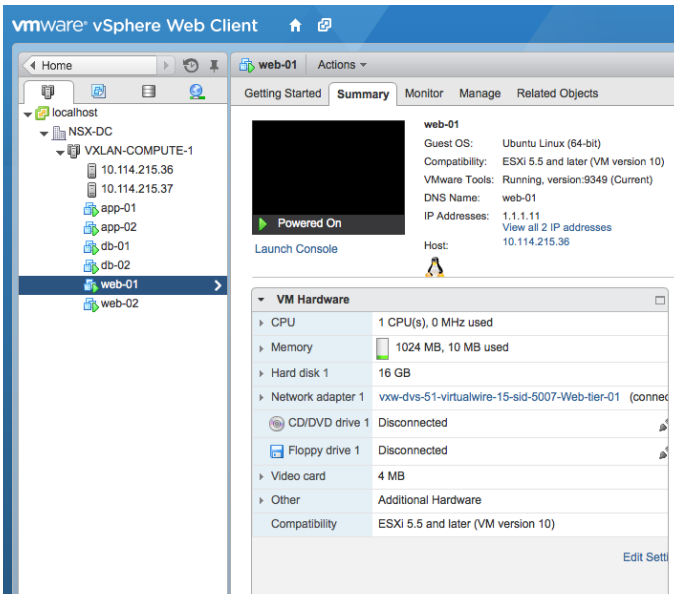


Figure 4 – LAB Topology View with vCenter.

For the purpose of this section, we are going to use Ubuntu Server VMs.

When installing the operating system, enable openSSH package for app-01/app-02 VMs. Enable the Tomcat package for web-01/web-02 VMs. This will activate SSH (port 22) and HTTP (port 8080) services on the respective servers.

To simulate the SQL service (TCP port 1433), we'll use the 'nc' utility provided natively in the Ubuntu distribution. The binary 'nc' is run listening to port 1433 on db-01/db-02 VMs.

Prerequisites

The following prerequisites apply to this scenario:

- The ESXi cluster must have been prepared in order to activate DFW, VXLAN and Logical Distributed Router modules. VTEP configuration must have been done as well:

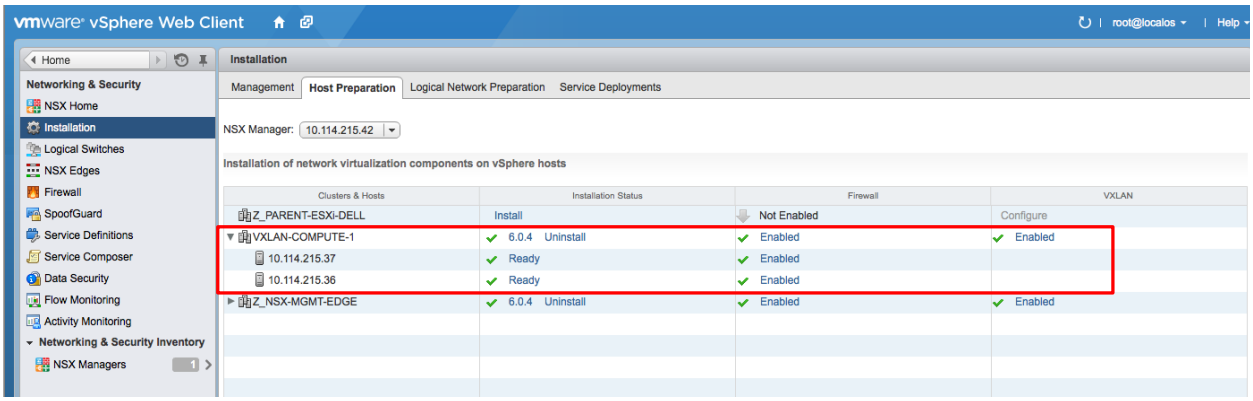


Figure 5 – Host Preparation Window.

- NSX Controllers must have been deployed in order to support VXLAN and Logical Distributed Router:

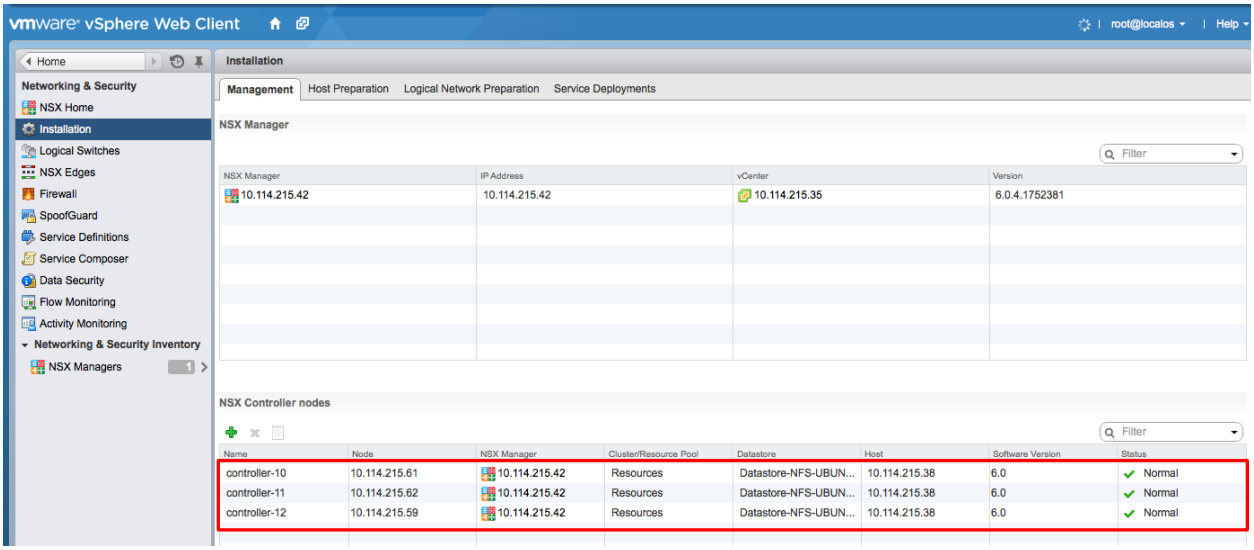


Figure 6 – NSX Controllers.

- VMTools must have been installed on all guest VMs:

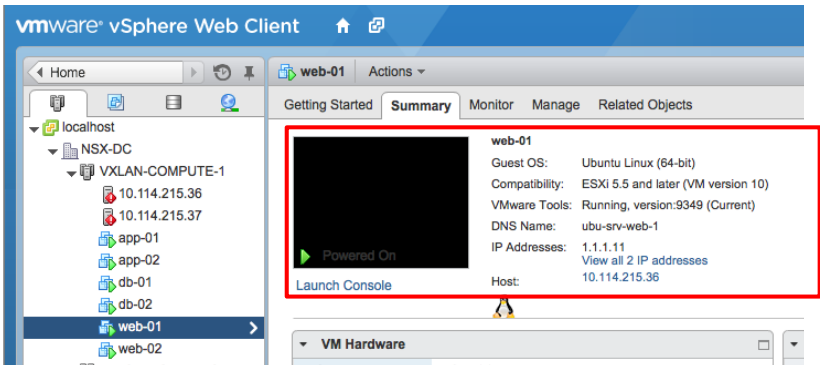


Figure 7 – VMTools Installed on Guest VM.

Please refer to the *NSX for vSphere Getting Started Guide* for more information.

Scenario 1: Microsegmentation for a three-tier Application using three L2 logical segments

Creating logical switches

In this section, you will create three Logical Switches (each provides a layer-2 logical segment to its attached VMs) and connect guest VMs to them:

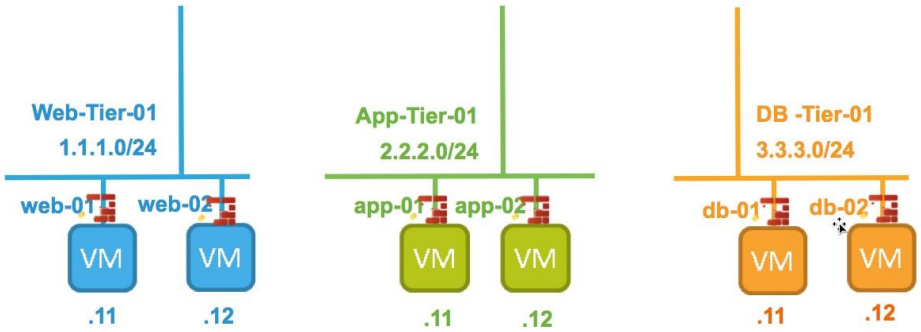


Figure 8 – Logical Switches and Guest VMs.

Step 1: Create three logical switches

From **NSX Home -> Logical Switches**, create the following three Logical Switches:

- Web-Tier-01
- App-Tier-01
- DB-Tier-01

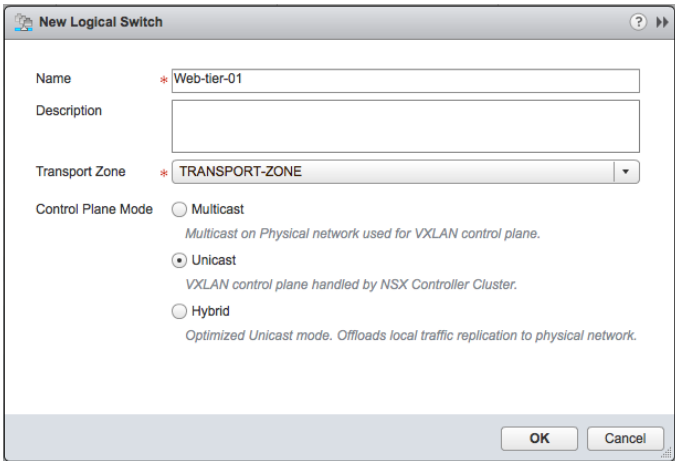


Figure 9 – Create a Logical Switch.

Step 2: Connect web/app/DB VMs to their respective logical switches

From **NSX Home -> Logical Switches**, add VMs to the appropriate logical switch:

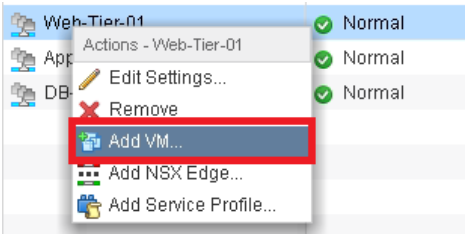


Figure 10 – Connect VMs to Logical Switch.

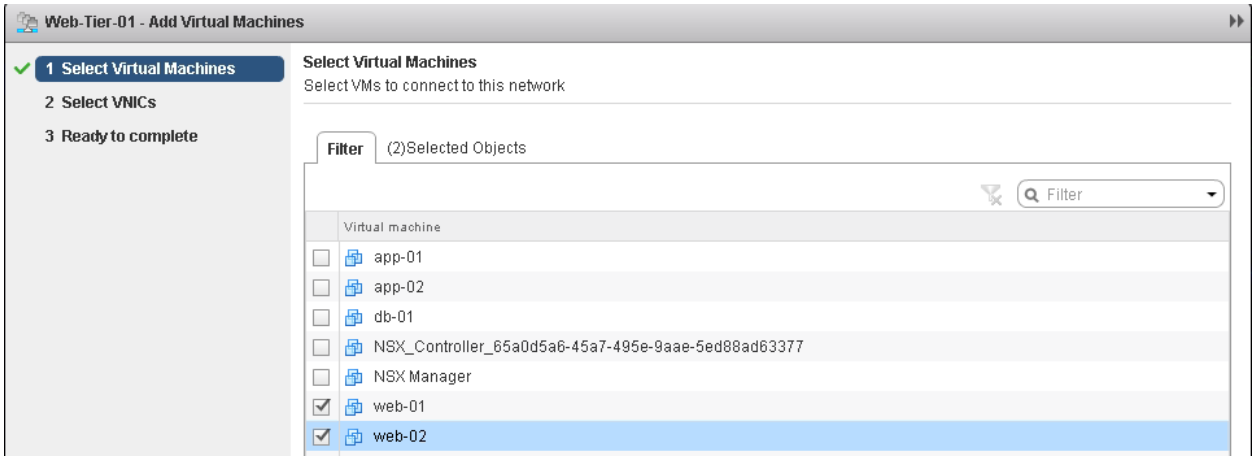


Figure 11 – Connect VMs to Logical Switch: Select VMs.

Select the vNIC for the VM to be attached to the logical switch and then click OK to finish the operation. You can check that each VM is connected to the intended Logical Switch using the VM summary tab.

From **vCenter Home -> Hosts and Clusters**, select the VM and look at the **Network adapter 1** field:

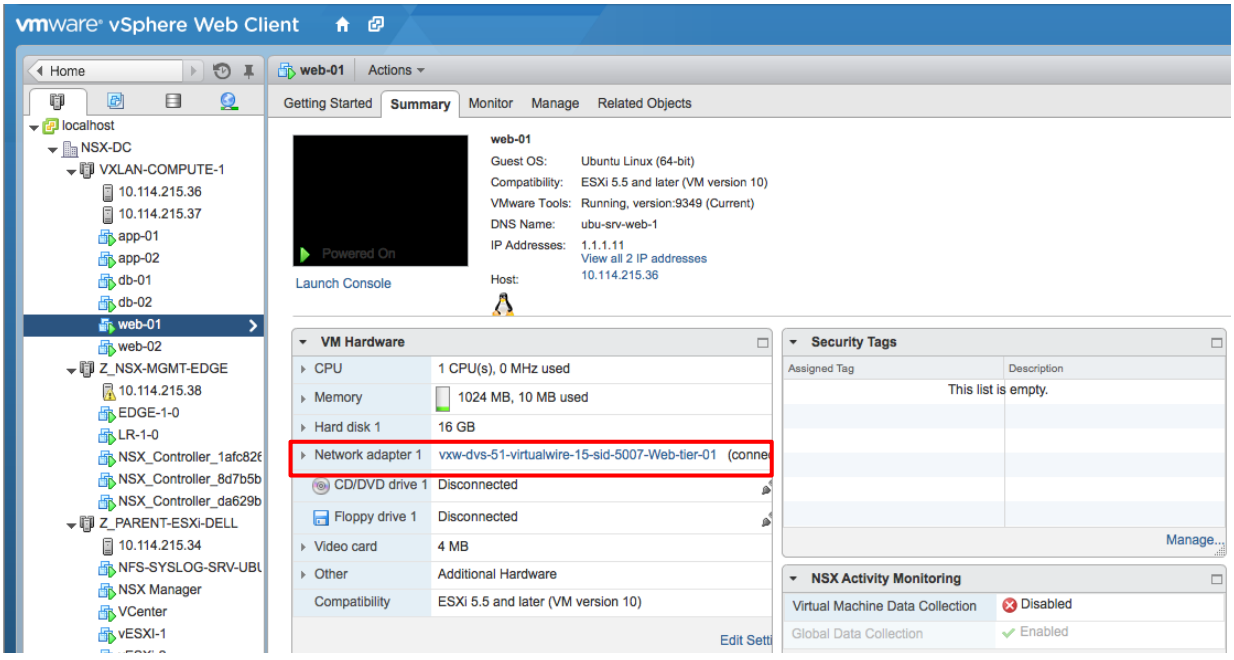


Figure 12 – Validate VM Network Adapter is Properly Connected to its L2 logical segment.

Step 3: Check that VMs on the same logical switch can communicate

Launch console access for web-01 VM and then try to ping web-02 from there:


```
root@web-01:~# ping 10.0.1.12
PING 10.0.1.12 (10.0.1.12) 56(84) bytes of data.
64 bytes from 10.0.1.12: icmp_req=1 ttl=64 time=12.9 ms
64 bytes from 10.0.1.12: icmp_req=2 ttl=64 time=0.711 ms
```

Figure 13 – Ping within Web Tier.

At this point, we've validated that the L2 logical segment is working across the two ESXi hosts. Keep in mind that the set of VMs for any given function (the web server layer in this case) is distributed across two physical servers.

Creating the Logical Distributed Router

In this section we will create the Logical Distributed Router (LDR) in order to enable communication between the three tiers:

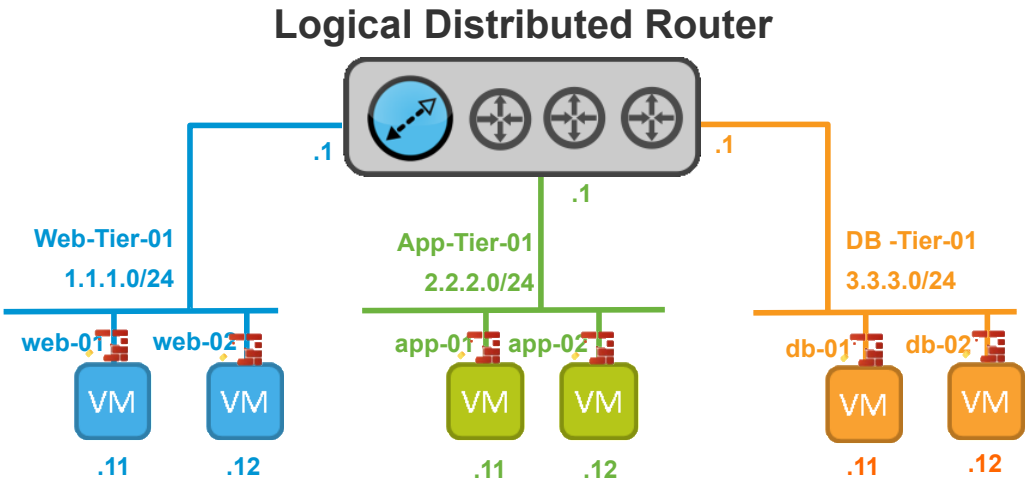


Figure 14 –Logical Distributed Router Instantiation.

By "logical *distributed* router," we mean that one instance of the Logical Router is deployed per ESXi host – at the data plane layer. With this type of logical router, when there is communication between two VMs that reside in two different application tiers and also happen to reside on the same ESXi host, those data packets remain on the host; there is no "hairpinning" effect.

Step 4: Validate the VMs on the same logical switch can communicate

From **NSX Home -> NSX Edges**, create a new instance of type "Logical Router".

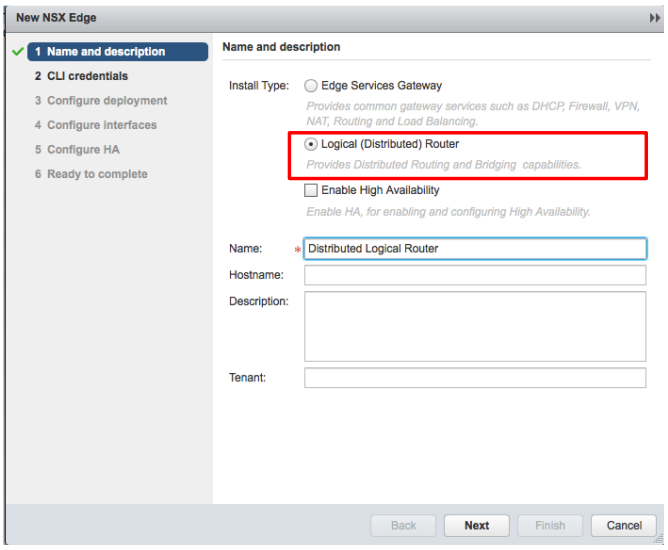


Figure 15 –Creating the Logical Distributed Router.

Please refer to the *NSX for vSphere Getting Started Guide* for more information about creating the LDR.

After creating the Logical Distributed Router, you should have this configuration applied to the interfaces:

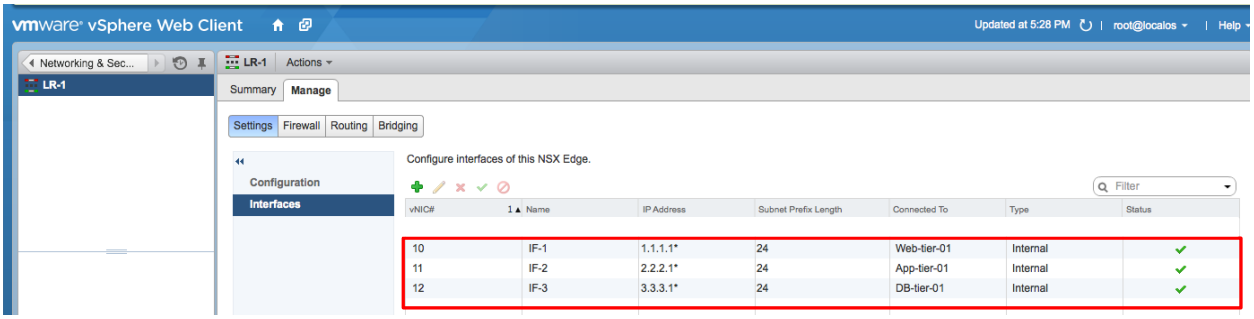


Figure 16 –Logical Distributed Router Interfaces Configuration.

Step 5: Validate that VMs on different logical switches (or tiers) can communicate

Launch console access for web-01 VM and then try to ping app-01 VM and db-02 VM from there:

```
root@web-01:~# ping 2.2.2.11
PING 2.2.2.11 (2.2.2.11) 56(84) bytes of data.
64 bytes from 2.2.2.11: icmp_req=1 ttl=63 time=0.901 ms
64 bytes from 2.2.2.11: icmp_req=2 ttl=63 time=0.353 ms
64 bytes from 2.2.2.11: icmp_req=3 ttl=63 time=0.680 ms
64 bytes from 2.2.2.11: icmp_req=4 ttl=63 time=0.247 ms
64 bytes from 2.2.2.11: icmp_req=5 ttl=63 time=0.252 ms
```

Figure 17 – Ping from Web Tier to App Tier.

```
root@web-01:~# ping 3.3.3.12
PING 3.3.3.12 (3.3.3.12) 56(84) bytes of data.
64 bytes from 3.3.3.12: icmp_req=1 ttl=63 time=2.18 ms
64 bytes from 3.3.3.12: icmp_req=2 ttl=63 time=0.381 ms
64 bytes from 3.3.3.12: icmp_req=3 ttl=63 time=0.411 ms
64 bytes from 3.3.3.12: icmp_req=4 ttl=63 time=0.394 ms
64 bytes from 3.3.3.12: icmp_req=5 ttl=63 time=0.412 ms
```

Figure 18 – Ping from Web Tier to DB Tier.

At this point of time, we have validated that the Logical Distributed Router is working fine within an ESXi host and across the two ESXi hosts.

Distributed Firewall (DFW) Configuration

Up to now, communications within a tier or across tiers are successful because default DFW policy rule is set to *Allow*. As a reminder, the DFW was activated as soon as the ESX cluster was prepared. Let's now start to configure the DFW security policies and verify their effects on traffic flows.

Step 6: Change DFW default policy rule from ALLOW to BLOCK

Objective of this step is to set the default firewall rule to "Block." After you complete this, your rules will be as follows:

Name	Source	Destination	Service	Action
Default Rule NDP	Any	Any	IPv6-ICMP Neighbor Solicitation IPv6-ICMP Neighbor Advertisement	Allow
Default Rule DHCP	Any	Any	DHCP-Server DHCP-Client	Allow
Default Rule	Any	Any	Any	Block

Go to **NSX Home -> Firewall** to visualize DFW security policy rules:

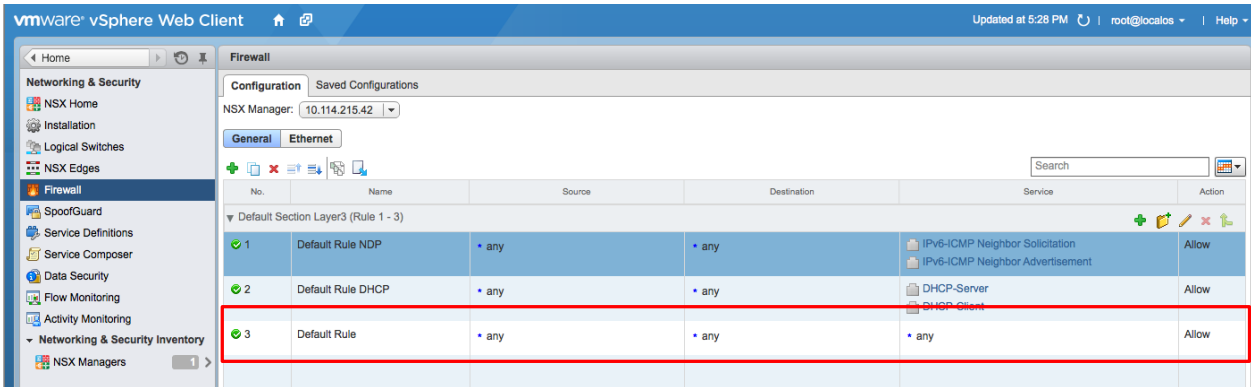


Figure 19 – DFW Default Policy Rule set to Allow.

Change the default policy rule from 'Allow' to 'Block' by clicking on the (+) button as indicated below:

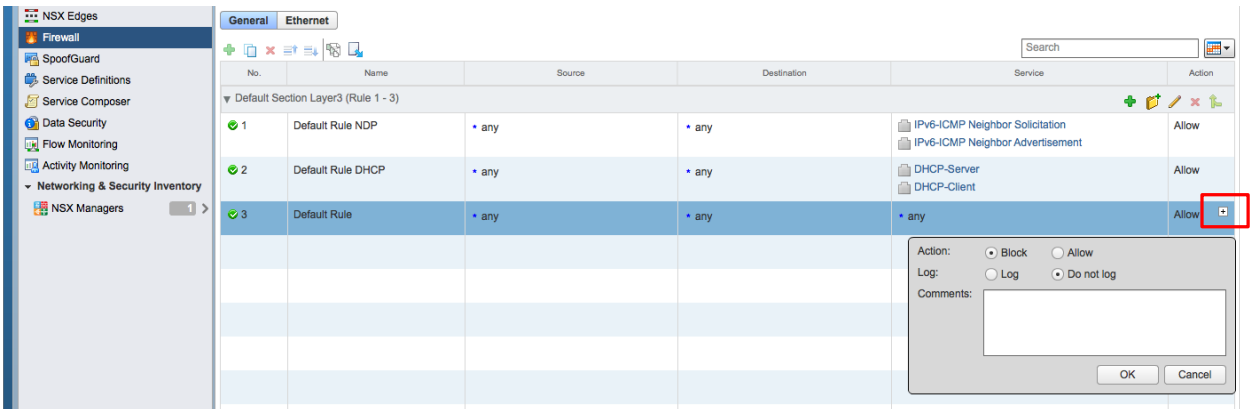


Figure 20 – Change DFW Default Policy Rule from Allow to Block.

Click on **Publish Changes** to force NSX Manager to push updated security policy rules down to all ESXi hosts in the cluster:

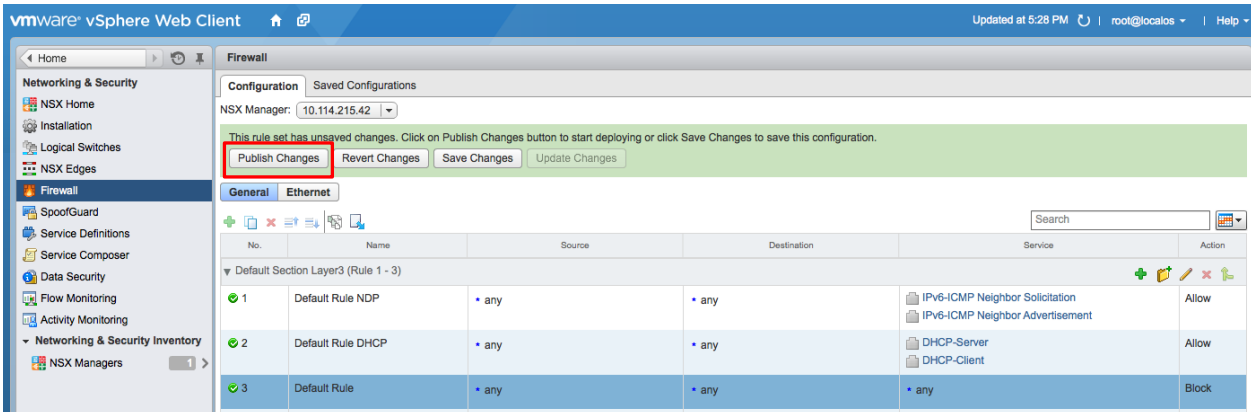


Figure 21 –Publish DFW Changes.

Step 7: Validate that the VMs cannot communicate

Launch console access for web-01 VM and then try to ping web-02, app-01 VM and db-02 VM from there:

```
root@web-01:~# ping 1.1.1.12
PING 1.1.1.12 (1.1.1.12) 56(84) bytes of data.
_
```

Figure 22 – Ping within Web Tier.

```
root@web-01:~# ping 2.2.2.11
PING 2.2.2.11 (2.2.2.11) 56(84) bytes of data.
_
```

Figure 23 – Ping from Web Tier to App Tier.

```
root@web-01:~# ping 3.3.3.12
PING 3.3.3.12 (3.3.3.12) 56(84) bytes of data.
_
```

Figure 24 – Ping from Web Tier to DB Tier.

As expected, changing the DFW's default policy rule from 'Allow' to 'Block' blocks all communications within the tier and across tiers. No VM can send a network packet to any other VM.

Step 8: Implement DFW security policy rules for intra-tier traffic

Objective of this step is to implement the following rules:

Source	Destination	Service	Action	Description
Any Web VM	Any Web VM	HTTP	Allow	Only allow HTTP traffic within Web tier
Any App VM	Any App VM	ICMP	Allow	Only allow ICMP traffic within App tier
Any DB VM	Any DB VM	ICMP	Allow	Only allow ICMP traffic within DB tier

Go to **NSX Home -> Firewall** to access DFW general window. Create a new section called 'INTRA-TIER' by clicking on the folder icon:

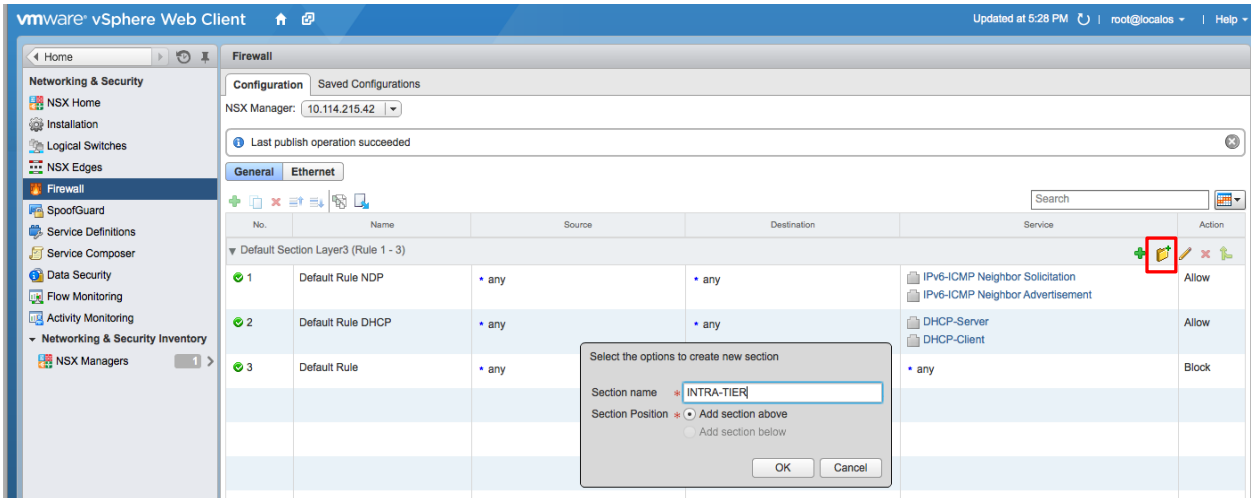


Figure 25 – Create New INTRA-TIER Section.

Create rules within the section by clicking on (+) button:

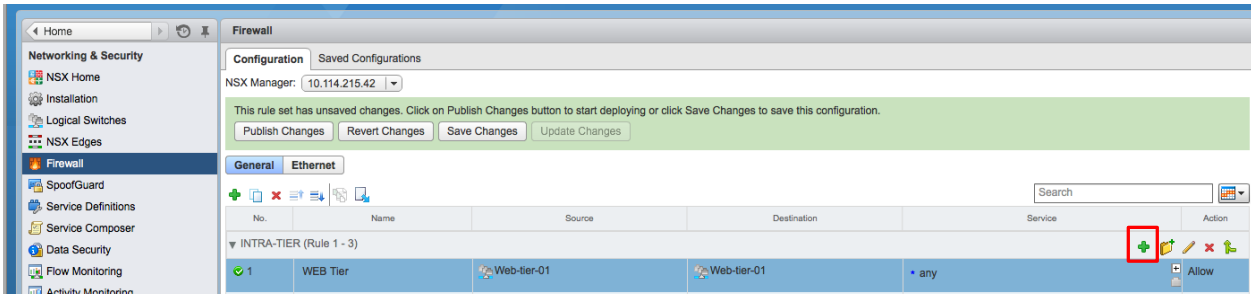


Figure 26 – Create New Rules within the Section.

For source and destination fields, use the logical switch object as shown below:

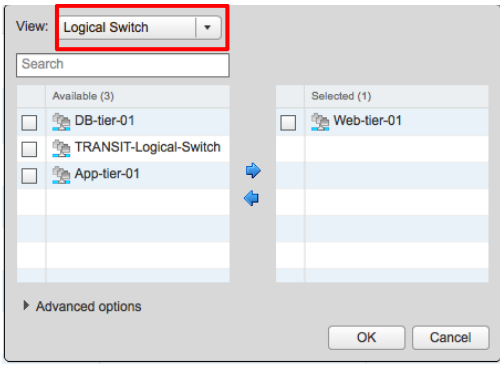


Figure 27 – Use Logical Switch for Source and Destination Fields.

To define service HTTP-8080 (port 8080), click on (+) button inside Service field:

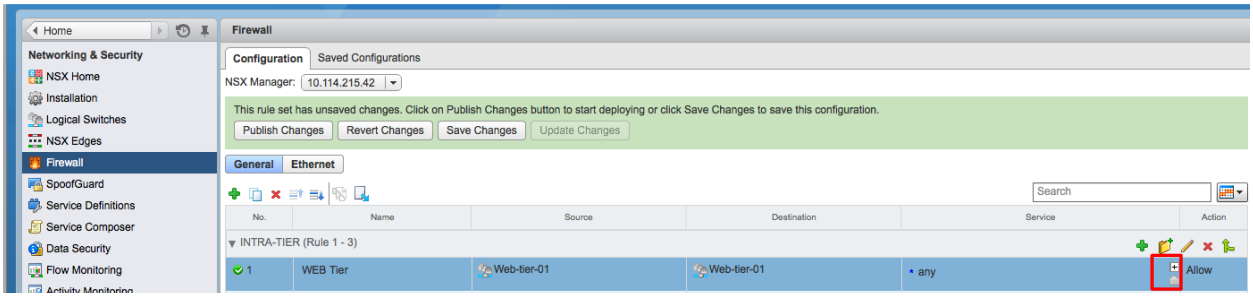


Figure 28 – Define new Service HTTP-8080 (1/3).

Select New -> Service:

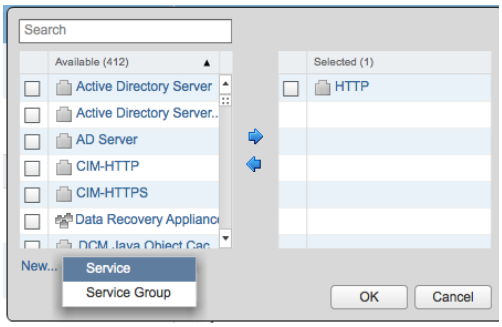


Figure 29 – Define new Service HTTP-8080 (2/3).

And fill the fields as shown below:

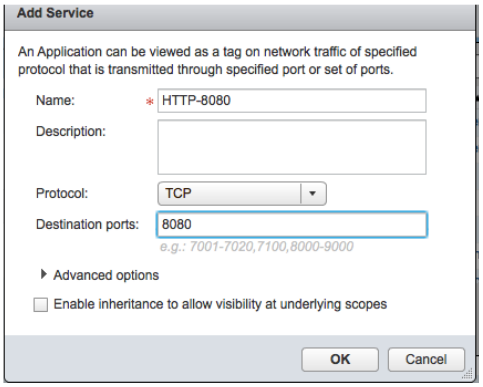


Figure 30 – Define new Service HTTP-8080 (3/3).

Click on OK button to create the new service HTTP-8080.

Create intra-tier rules for App and DB tiers.

You should obtain this security policy table at the end of the configuration:

Name	Source	Destination	Service	Action	Description
WEB Tier	Any Web VM	Any Web VM	HTTP-8080	Allow	Allow web traffic within Web tier
APP Tier	Any App VM	Any App VM	ICMP Echo Reply ICMP Echo	Allow	Allow ICMP traffic within App tier
DB Tier	Any DB VM	Any DB VM	ICMP Echo Reply ICMP Echo	Allow	Allow ICMP traffic within DB tier

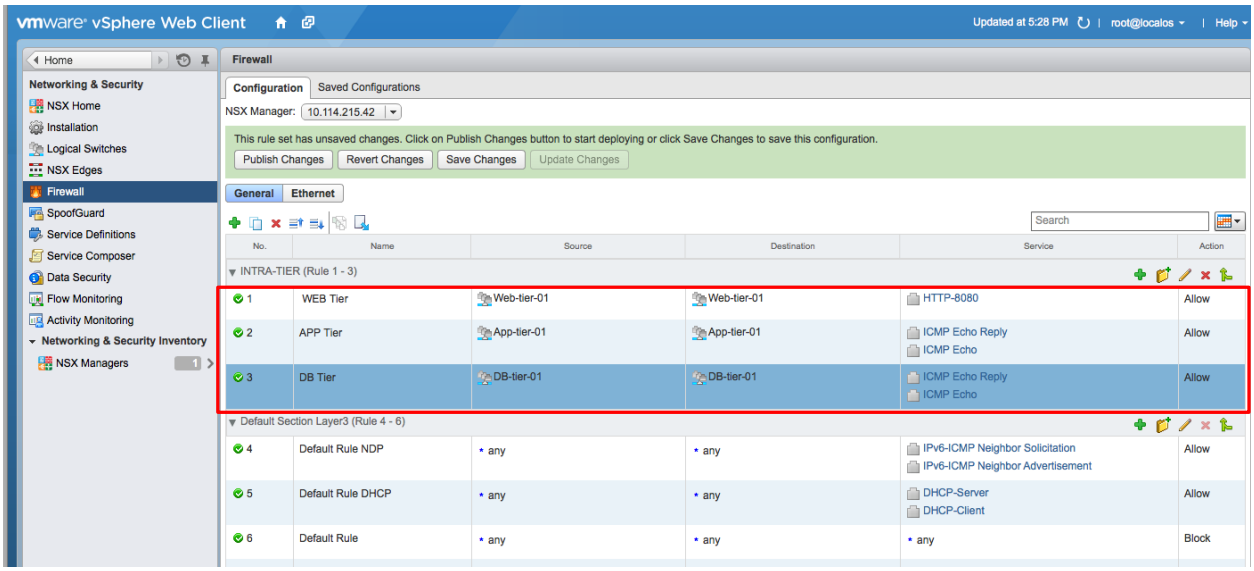


Figure 31 – Complete Security Policy Definition for Intra-Tier.

Finally click the **Publish Changes** button to enforce new security policy definition.

Step 9: Validate intra-tier VM communication

Launch console access for web-01 VM and then perform a HTTP GET request to web-02 VM using curl utility:

```
# curl http://1.1.1.12:8080/index.html
```

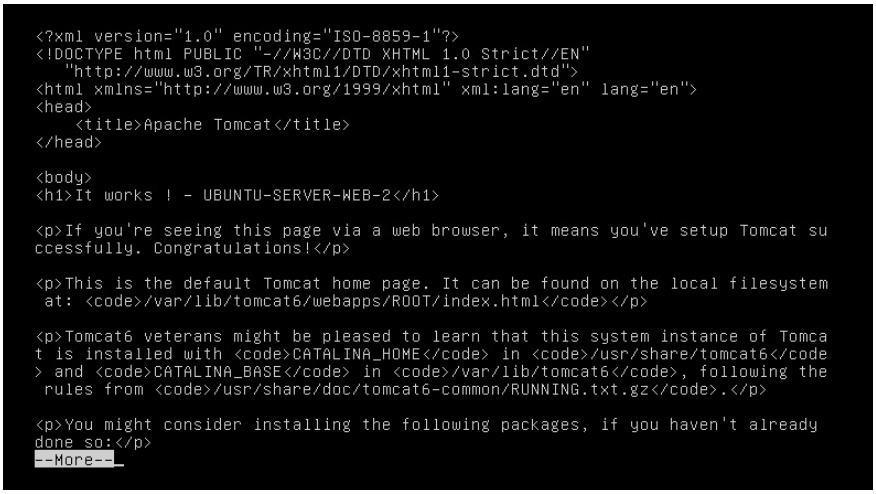


Figure 32 – Web Tier Intra Communication.

Launch console access for app-01 VM and then perform a ping to app-02 VM:


```
root@app-01:~# ping 2.2.2.12
PING 2.2.2.12 (2.2.2.12) 56(84) bytes of data.
64 bytes from 2.2.2.12: icmp_req=1 ttl=64 time=0.448 ms
64 bytes from 2.2.2.12: icmp_req=2 ttl=64 time=0.361 ms
64 bytes from 2.2.2.12: icmp_req=3 ttl=64 time=0.458 ms
64 bytes from 2.2.2.12: icmp_req=4 ttl=64 time=0.396 ms
```

Figure 33 – App Tier Intra Communication.

Launch console access for db-01 VM and then perform a ping to db-02 VM:

```
root@db-01:~# ping 3.3.3.12
PING 3.3.3.12 (3.3.3.12) 56(84) bytes of data.
64 bytes from 3.3.3.12: icmp_req=1 ttl=64 time=0.483 ms
64 bytes from 3.3.3.12: icmp_req=2 ttl=64 time=0.419 ms
64 bytes from 3.3.3.12: icmp_req=3 ttl=64 time=0.431 ms
64 bytes from 3.3.3.12: icmp_req=4 ttl=64 time=0.342 ms
64 bytes from 3.3.3.12: icmp_req=5 ttl=64 time=0.429 ms
```

Figure 34 – DB Tier Intra Communication.

Step 10: Implement DFW security policy rules for inter-tier traffic

Objective of this step is to implement the following rules:

Source	Destination	Service	Action	Description
Any Web VM	Any App VM	SSH	Allow	Only allow SSH from Web tier to App tier
Any App VM	Any DB VM	TCP port 1433	Allow	Only allow TCP port 1433 from App tier to DB tier

Create a new section called INTER-TIER and then create above rules using the logical switch object for source and destination field; Create new service definition for TCP-1433 (refer to step 8 if needed).

You should have the following rules in your policy rules table at the end of the configuration:

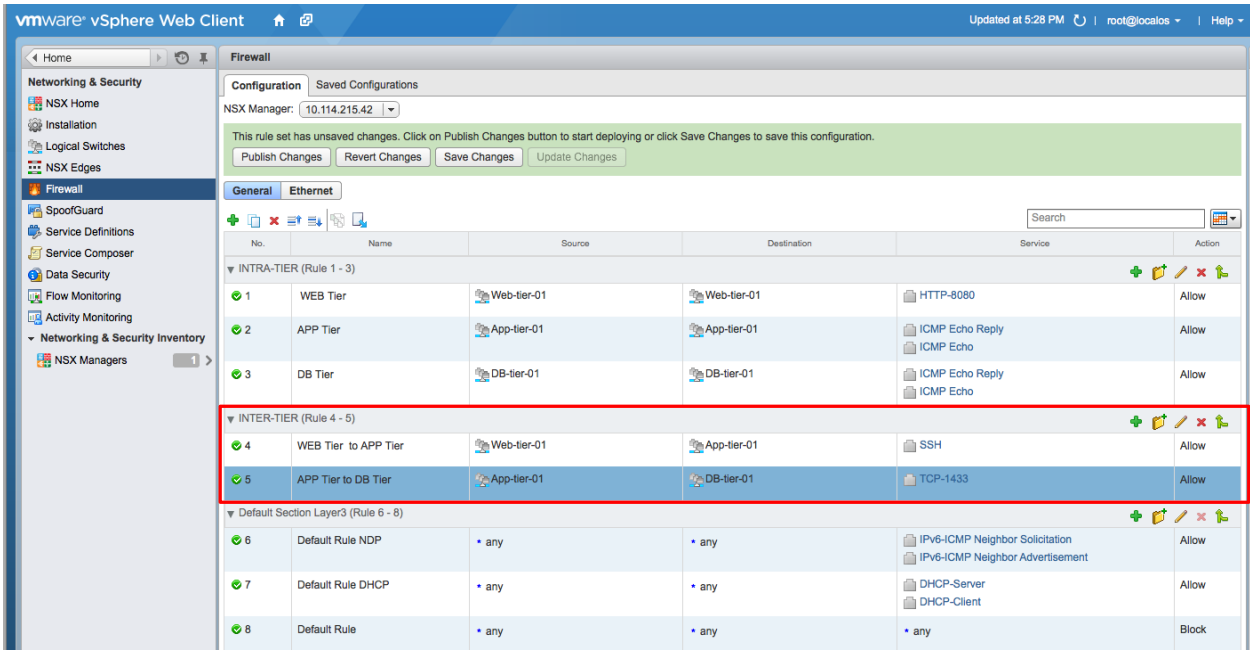


Figure 35 – DFW Policy Rules for Inter-Tier Traffic.

Click the 'Publish Changes' button to enforce new security policy definition.

Step 11: Validate inter-tier VM communication

Launch console access for web-01 VM and then perform a SSH request to app-02 VM:

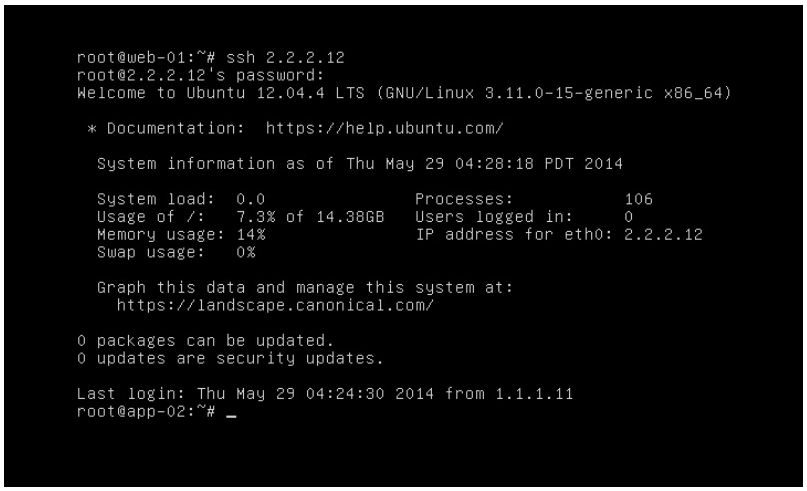


Figure 36 – SSH from Web Tier to App Tier.

Launch console access for app-01 VM and then perform a TCP-1433 request to db-02 VM. To simulate the service on db-02, we are going to use the *nc* utility listening to port TCP 1433:

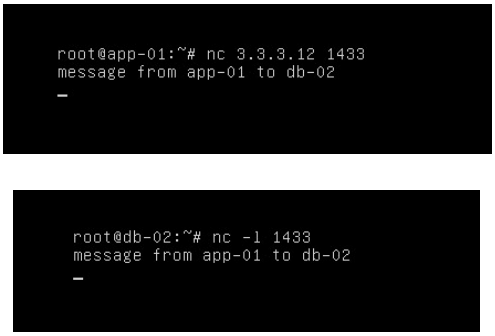


Figure 37 – Communication from App Tier to DB Tier on port TCP 1433.

App-01 VM was able to send a message to db-01 VM using destination TCP port 1433. This concludes our first scenario.

Note! Is it not necessary to delete anything before you begin working on Scenario 2.

Scenario 2: Microsegmentation for a three-tier application using a single L2 logical segment

Logical Switch Instantiation and VM Connectivity

In this section, you will create one Logical Switch connect all guest VMs to it.

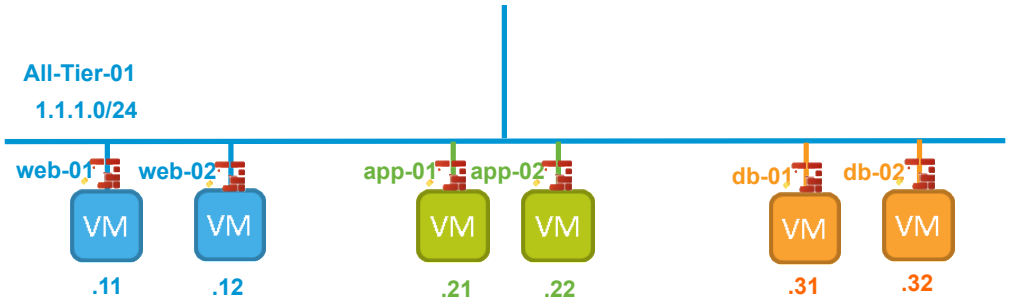


Figure 38 – Logical Switch and Guest VMs.

Note that the VM IP addresses have changed compared with *Scenario 1* because all VMs now reside in a single broadcast domain (that is, in the same subnet). Modify the VMs' IP addresses in your lab to reflect the new address scheme.

Step 1: Create the logical switch

From **NSX Home -> Logical Switches**, create the following Logical Switch:

- All-tier-01

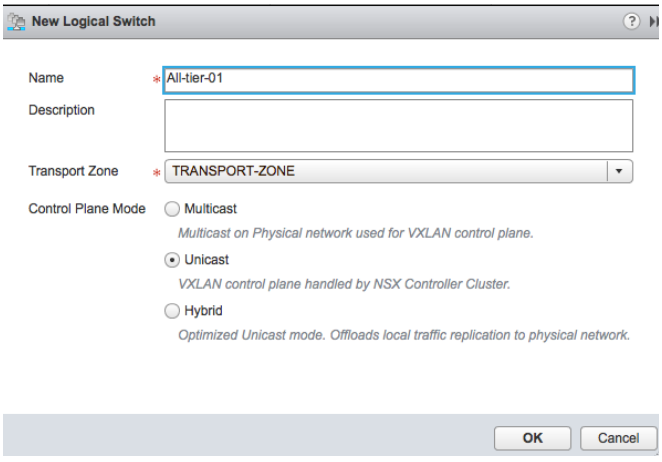


Figure 39 – Create a Logical Switch.

Step 2: Connect web/app/DB VMs to the logical switch

From **NSX Home -> Logical Switches**, select All-tier-01 logical switch and click on Actions -> Add VM

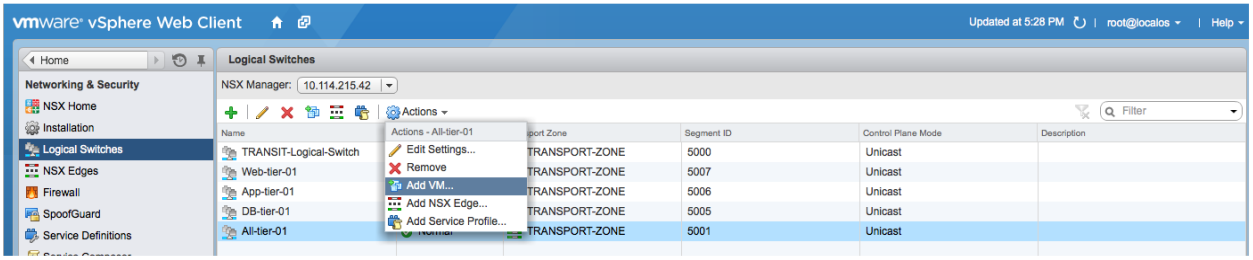


Figure 40 – Connect VMs to Logical Switch.

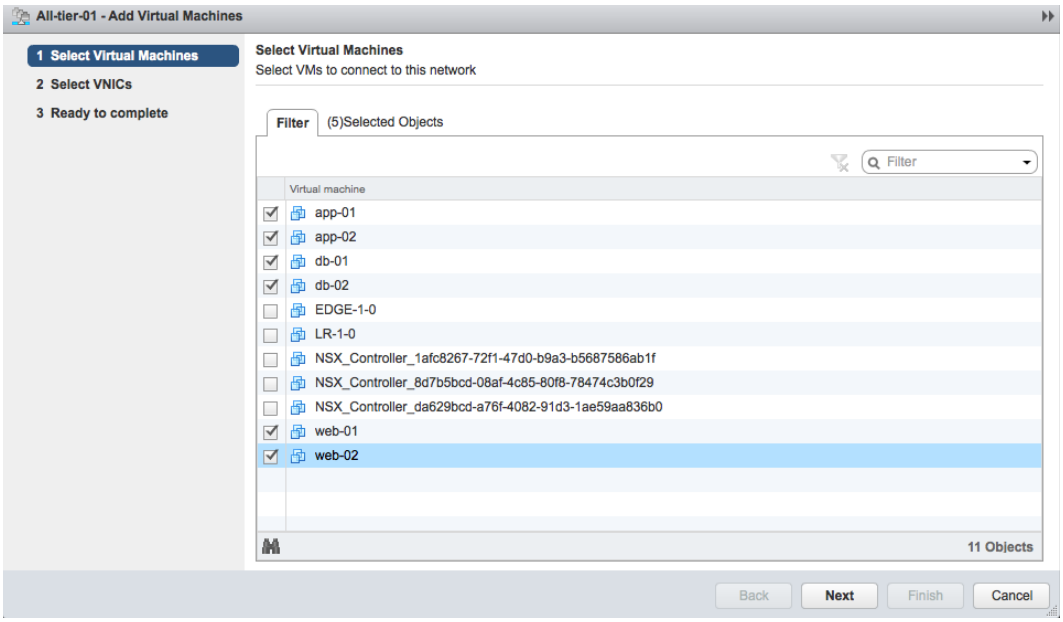


Figure 41 – Select all VMs to be Connected to Logical Switch.

Select the vNIC for the VM to be attached to the logical switch and then click on OK to finish the operation.

Step 3: Validate that the VMs on the logical switch cannot communicate with one another

Because all guest VMs are now connected to a new logical switch (compared with Scenario 1), the DFW's default rule applies to them. Since the default rule has been set to 'Block', no communication is possible.

Launch console access for web-01 VM and then try to ping web-02, app-01, app-02, db-01 and db-02:

```
root@web-01:~# ping 1.1.1.12
PING 1.1.1.12 (1.1.1.12) 56(84) bytes of data.
^C
--- 1.1.1.12 ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5039ms

root@web-01:~# _
```



```
root@web-01:~# ping 1.1.1.21
PING 1.1.1.21 (1.1.1.21) 56(84) bytes of data.
^C
--- 1.1.1.21 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3022ms

root@web-01:~# _
```

```
root@web-01:~# ping 1.1.1.31
PING 1.1.1.31 (1.1.1.31) 56(84) bytes of data.
^C
--- 1.1.1.31 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3023ms

root@web-01:~# _
```

Figure 42 – Ping from web-01 to other VMs.

Creating the Security Groups

You will now create three Security Groups to map VMs of same function together:

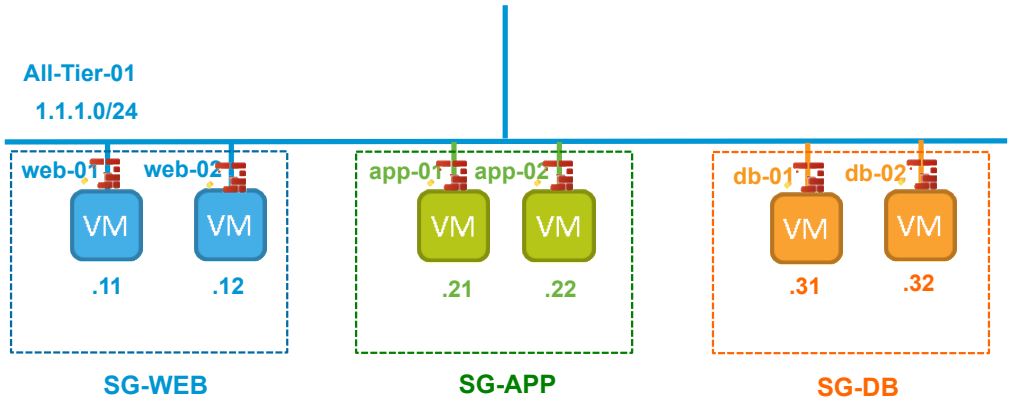


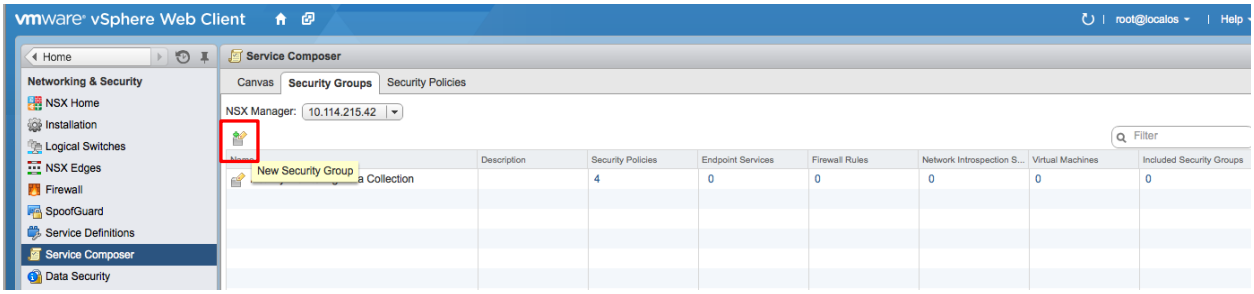
Figure 43 – Security Groups Applied per VM Function.

Step 4: Create three Security Groups: SG-WEB, SG-APP, SG-DB.

From NSX Home -> Service Composer -> Security Groups, create the following Security Groups:

- SG-WEB: static inclusion with web-01 and web-02 VM
- SG-APP: static inclusion with app-01 and app-02 VM
- SG-DB: static inclusion with db-01 and db-02 VM

Click on the 'New Security Group' icon:



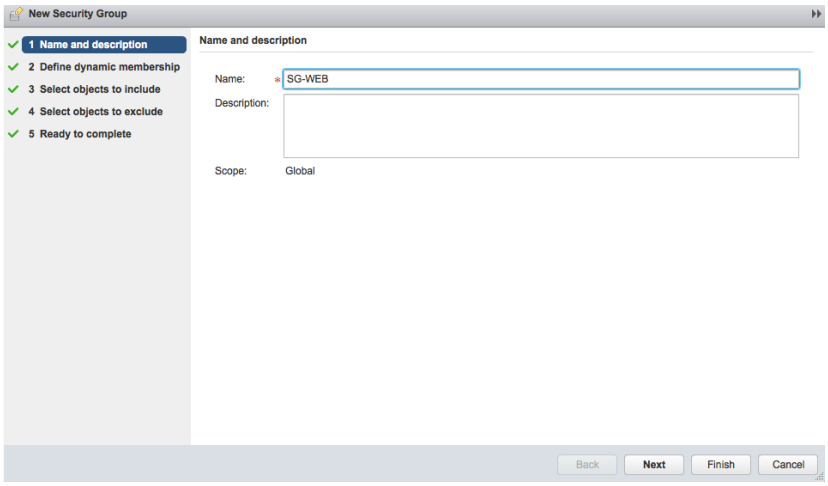


Figure 44 – Add new Security Group.

Click Next and then Next again to jump to 'Select objects to include'. Click the 'Virtual Machine' tab and select appropriate VM:

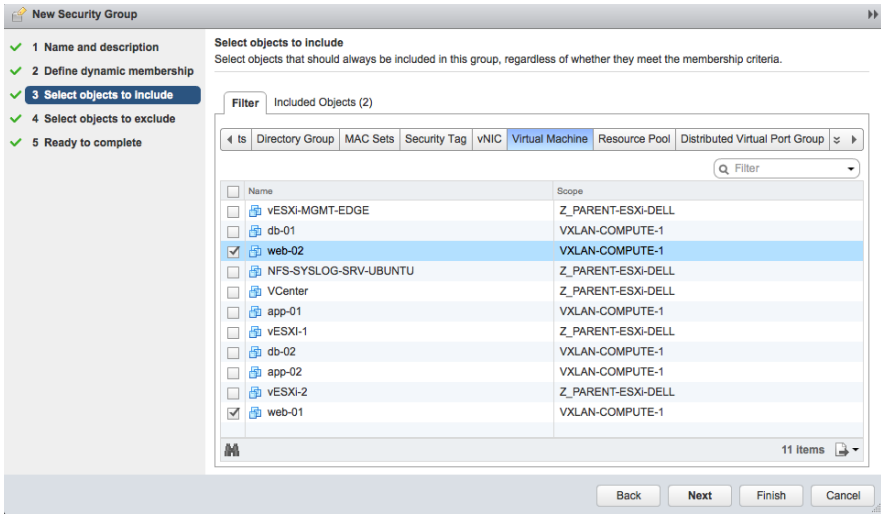


Figure 45 – Select the VMs to be included

Click on Finish to create the Security Group.

At the end of the configuration, you should see the following Security Groups:

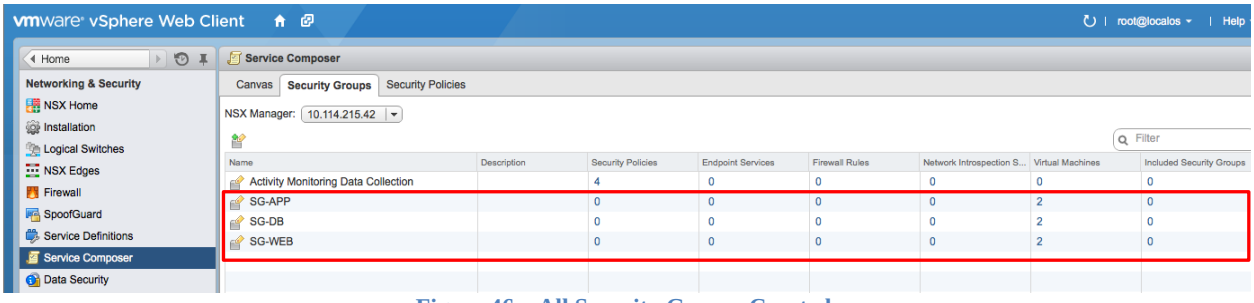


Figure 46 – All Security Groups Created.

To verify VMs mapped into a Security Group, click on the number link in the Virtual Machines column:

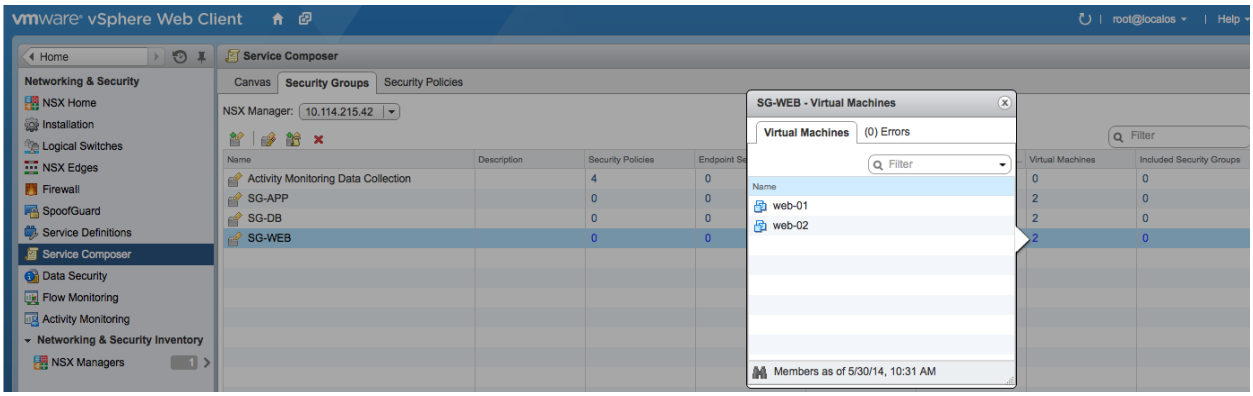


Figure 47 – Verify VMs Mapped into Security Group.

Configuring the Distributed Firewall (DFW)

In this section, you will configure DFW security policies and verify their effects on traffic flows. Since all VMs belong to one tier now, we will use 'function' terminology to differentiate among the VMs: *web-01/ web-02* serve the WEB function, *app-01 / app-02* serve the APP function and *db-01/db-02* serve the DB function. In this context, "intra-function" refers to network traffic within a function, and "inter-function" refers to network traffic across functions.

Step 5: Implement DFW security policy rules for intra-function traffic

Objective of this step is to implement the following rules:

Source	Destination	Service	Action	Description
Any Web VM	Any Web VM	HTTP	Allow	Only allow HTTP traffic within Web function
Any App VM	Any App VM	ICMP	Allow	Only allow ICMP traffic within App function
Any DB VM	Any DB VM	ICMP	Allow	Only allow ICMP traffic within DB function

Go to **NSX Home -> Firewall** to access DFW general window.

Create a new section called 'INTRA-FUNCTION by clicking the folder icon:

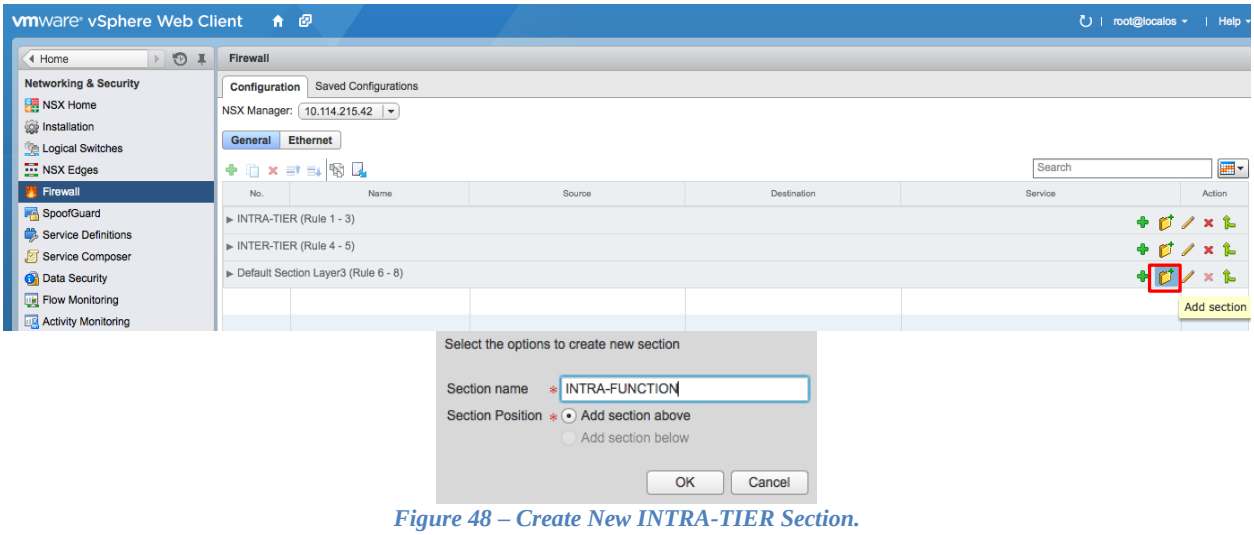


Figure 48 – Create New INTRA-TIER Section.

Create rules within the section by clicking the plus-sign (+) button:

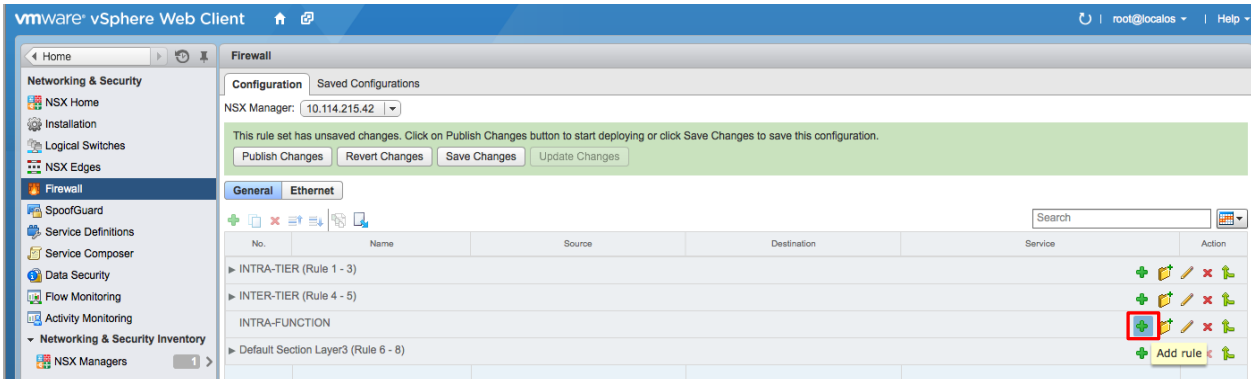


Figure 49 – Create New Rules within the Section.

For source and destination fields, use the Security Groups object as shown below:

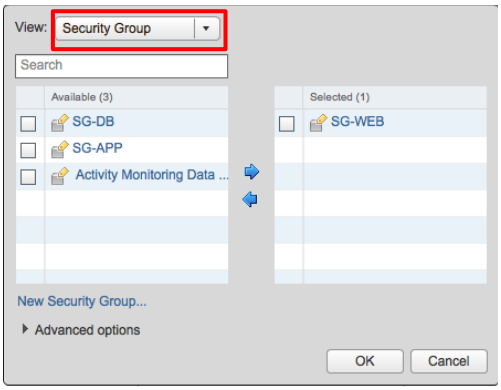


Figure 50 – Use Security Groups for Source and Destination Fields.

You should see this security policy table at the end of the configuration:

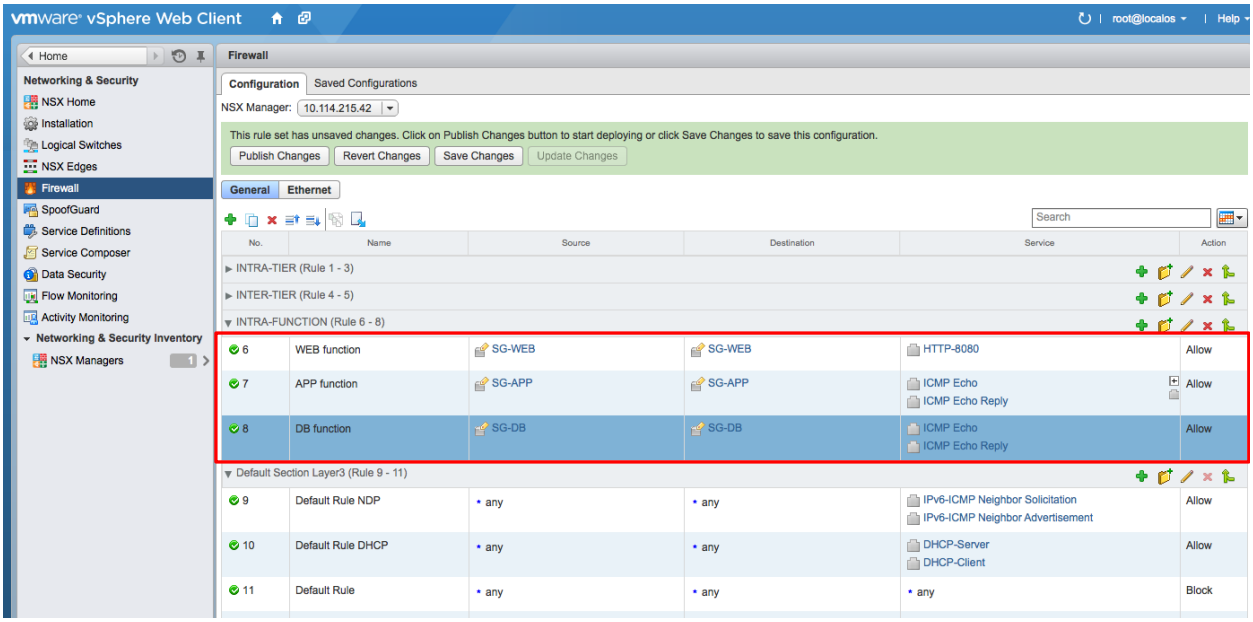


Figure 51 – Complete Security Policy Definition for Intra-Function.

(Use the service definition for HTTP-8080, which you created in *Scenario 1*.)

Finally click ‘Publish Changes’ button to enforce new security policy definition.

Step 6: Validate intra-function VM communication

Launch console access for web-01 VM and then perform a HTTP GET request to web-02 VM using curl utility:

```
# curl http://1.1.1.12:8080/index.html
```

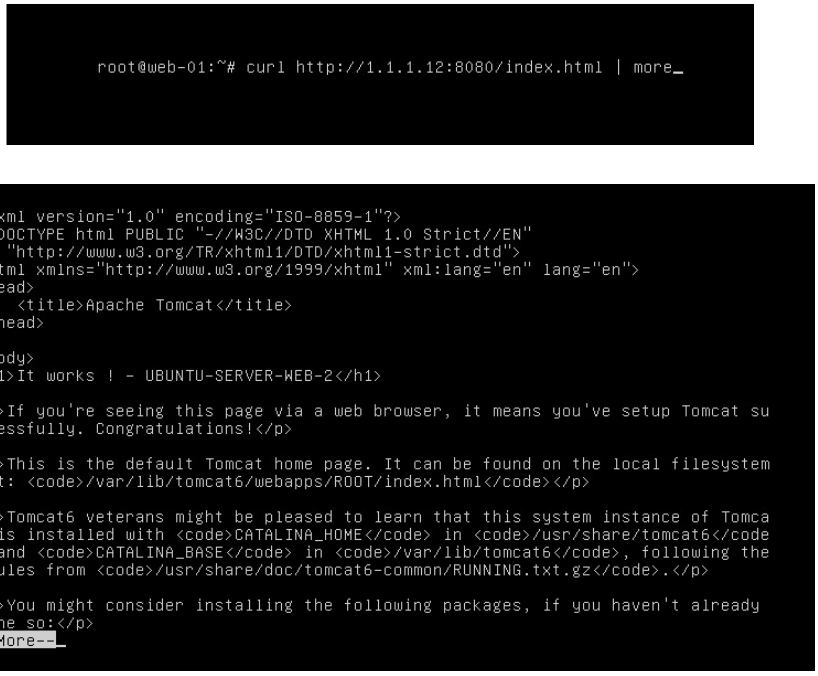


Figure 52 – Web Function Intra Communication.

Launch console access for app-01 VM and then perform a ping to app-02 VM:

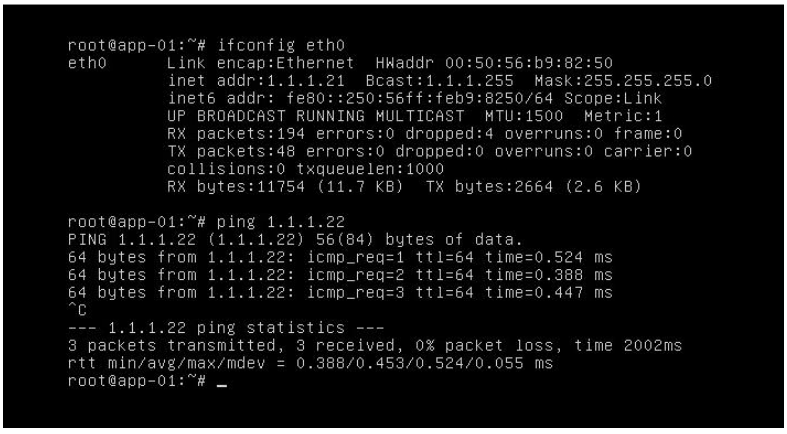


Figure 53 – App Function Intra Communication.

Launch console access for db-01 VM and then perform a ping to db-02 VM:


```
root@db-01:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:50:56:b9:a4:ed
          inet addr:1.1.1.31  Bcast:1.1.1.255  Mask:255.255.255.0
          inet6 addr: fe80::250:56ff:feb9:a4ed/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:139 errors:0 dropped:5 overruns:0 frame:0
          TX packets:39 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8340 (8.3 KB)  TX bytes:1950 (1.9 KB)

root@db-01:~# ping 1.1.1.32
PING 1.1.1.32 (1.1.1.32) 56(84) bytes of data.
64 bytes from 1.1.1.32: icmp_req=1 ttl=64 time=16.1 ms
64 bytes from 1.1.1.32: icmp_req=2 ttl=64 time=0.402 ms
64 bytes from 1.1.1.32: icmp_req=3 ttl=64 time=0.426 ms
64 bytes from 1.1.1.32: icmp_req=4 ttl=64 time=0.398 ms
64 bytes from 1.1.1.32: icmp_req=5 ttl=64 time=0.436 ms
^C
--- 1.1.1.32 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4004ms
rtt min/avg/max/mdev = 0.398/3.554/16.112/6.279 ms
root@db-01:~# _
```

Figure 54 – DB Function Intra Communication.

Step 7: Implement DFW security policy rules for inter-function traffic

The objective of this step is to implement the following rules:

Source	Destination	Service	Action	Description
Any Web VM	Any App VM	SSH	Allow	Only allow SSH from Web tier to App function
Any App VM	Any DB VM	TCP port 1433	Allow	Only allow TCP port 1433 from App tier to DB function

Create a new section called INTER-FUNCTION and then create the above rules using a Security Group object for the source and destination field; Use the service definition for TCP-1433 previously created in Scenario 1. You should see this type of policy rules table at the end of the configuration:

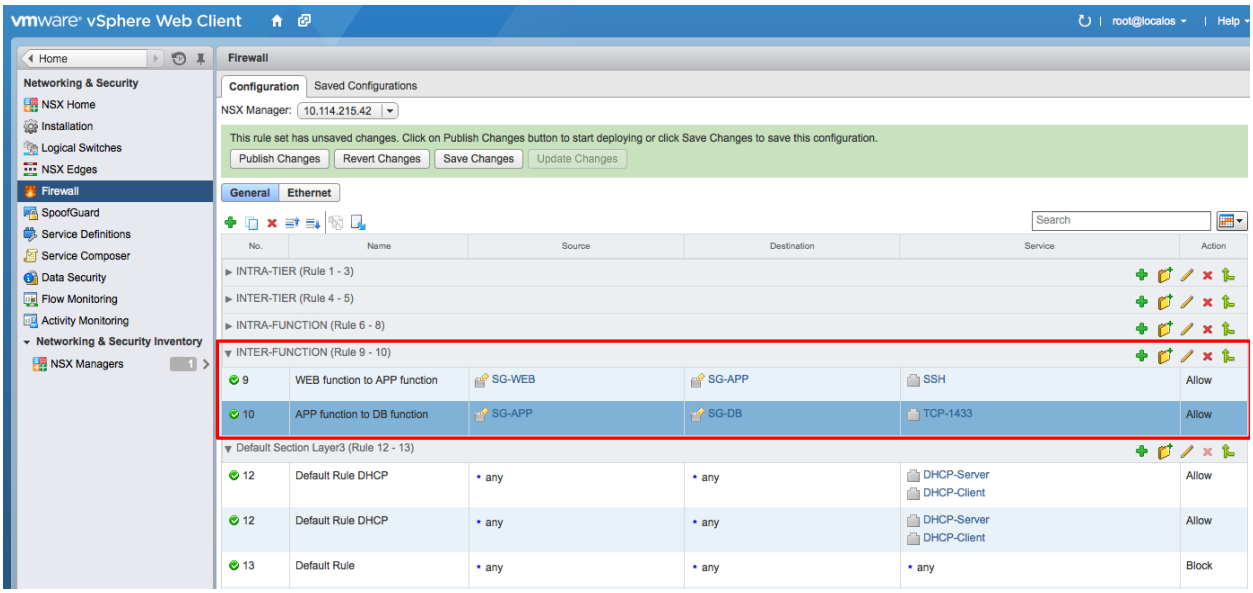


Figure 55 – DFW Policy Rules for Inter-Function Traffic.

Click on 'Publish Changes' button to enforce new security policy definition.

Step 8: Validate inter-function VM communication.

Launch console access for web-01 VM and then perform a SSH request to app-02 VM:

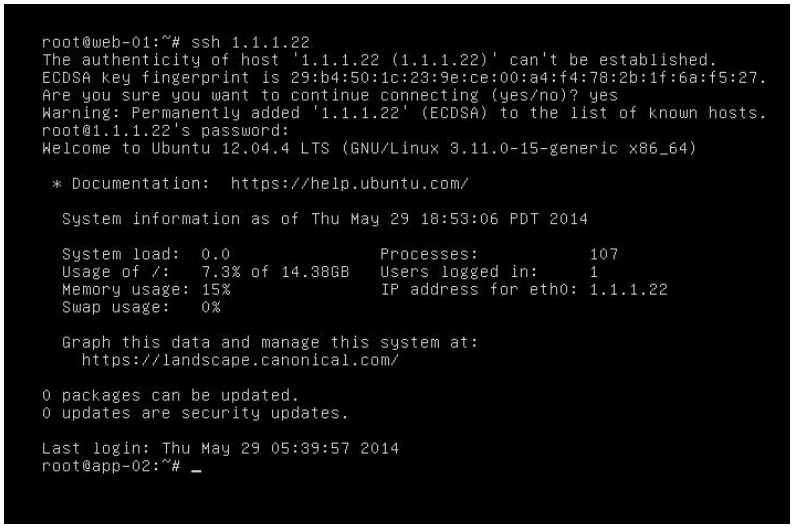


Figure 56 – SSH from Web Function to App Function.

Launch console access for app-01 VM and then perform a TCP-1433 request to db-02 VM. To simulate the service on db-02, we are going to use the *nc* utility listening to port TCP 1433:

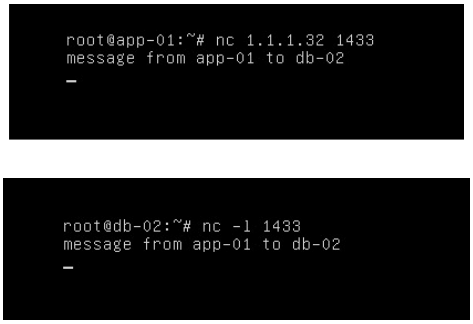


Figure 57 – Communication from App Tier to DB Tier on port TCP 1433.

The app-01 VM was able to send successfully a message to db-01 VM using destination TCP port 1433. This concludes our second scenario.

Conclusion: Microsegmentation

In this document, we have seen two ways to use the microsegmentation capabilities provided by VMware NSX, first as a three-tier application using *three separate L2 logical segments*, and second as a three-tier application using *a single L2 logical segment*. We summarize both below.

Three-tier application using three separate L2 logical segments

In *Scenario 1*, each tier is placed on its own L2 logical network segment by connecting its VMs to a dedicated logical switch for that tier. A logical switch represents an L2 logical segment (L2 broadcast domain) that you can extend across all physical ESXi hosts, meaning we can place the VMs anywhere in the ESXi cluster, and they will still have layer-2 connectivity to the other VMs in that tier.

The Logical Distributed Router enables connectivity among the three tiers. Because the logical router function is distributed, when there is traffic between two VMs from different tiers but residing on the same ESXi host, that traffic remains local to the host.

Finally, to enforce network traffic filtering within a tier or between tiers, the distributed firewall (DFW) is used. Because our security policy rules identify traffic based on the source logical switch and destination logical switch of the packets, it is easy to allow or block traffic flows within or between the tiers. This capability provides easy-to-read and easy-to-maintain rules, and it optimizes the number of needed rules.

Three-tier application using a single L2 logical segment

In *Scenario 2*, we use "function" terminology to group VMs of the same role together (for example, web-01/web-02 belong to the WEB function), so that they are represented by the same Security Group object. In the security policy rule, we specify the Security Group name as a parameter in the source or destination field. This provides the same operational efficiency and flexibility we saw in *Scenario 1*.

Getting Help and More Information

NSX-v Documentation

In addition to this document, you can read the following documents for help setting up NSX-v. All are available from https://www-stage.vmware.com/support/pubs/nsx_pubs.html:

- NSX for vSphere Installation and Upgrade Guide
- NSX for vSphere Administration Guide
- NSX for vSphere API Reference Guide
- NSX for vSphere Command Line Interface Reference

Contacting the NSX Technical Services Team

You can reach the NSX technical services team at http://www.vmware.com/support.html.