System Administration

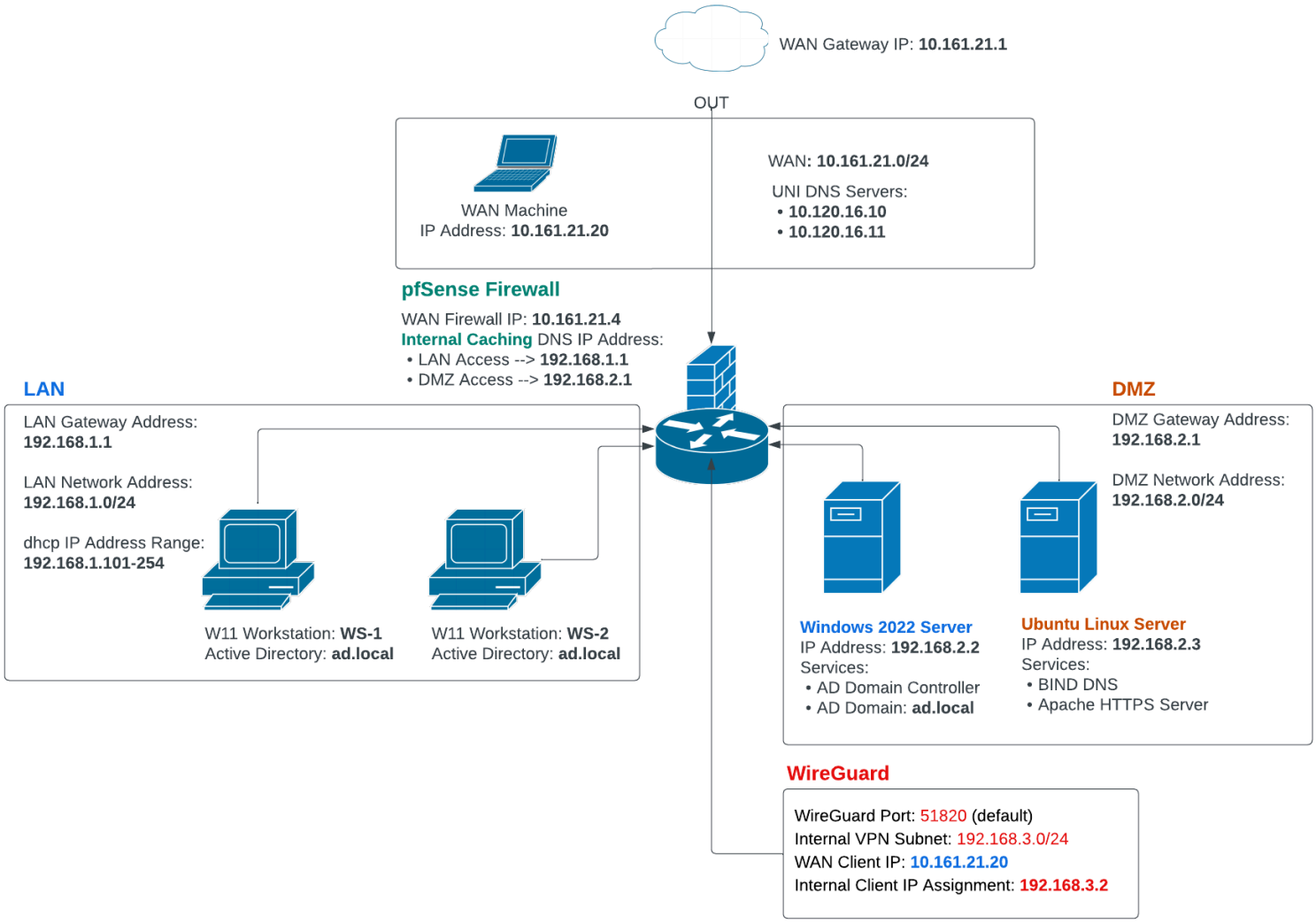
Final Documentation

Created by: Dominick Olhava

Table of Contents

Network Overview

Machine	Service(s)	IP Address	FQDN
pfSense Firewall	- Internal Caching DNS - WireGuard VPN	10.161.21.4	pfsense.slick.firewall
Windows Server	- AD Domain Controller - AD Domain: **ad.local**	192.168.2.2	slickbrickcentral.com (web server) ubuntu.slickbrickcentral.com
Ubuntu Server	- BIND DNS - Apache Web Server	192.168.2.3	win.slickbrickcentral.com



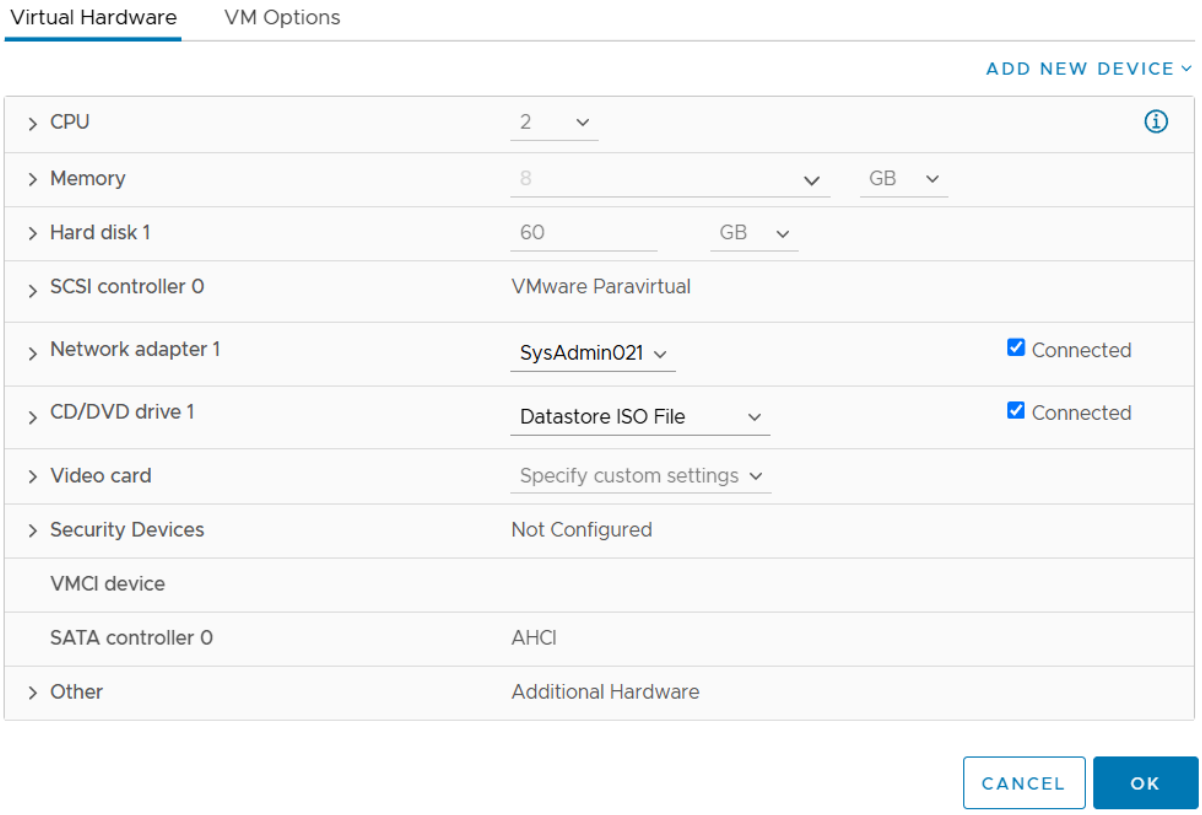
Project 1: Linux Server

Below contains steps outlining the configuration of an **Ubuntu Linux Server** that will be used to host a Static HTTPS Web Server and BIND for Internal DNS mappings. The Linux Server will start in the WAN under the Static IP Address: **10.161.21.2**.

VM Configuration

Guest Os: Linux (64 Bit)
Hardware: 2 CPUs, 8 GB RAM, Thin Provision, Use Datastore ISO file
ISO file: **Ubuntu 24.04 Live Server**

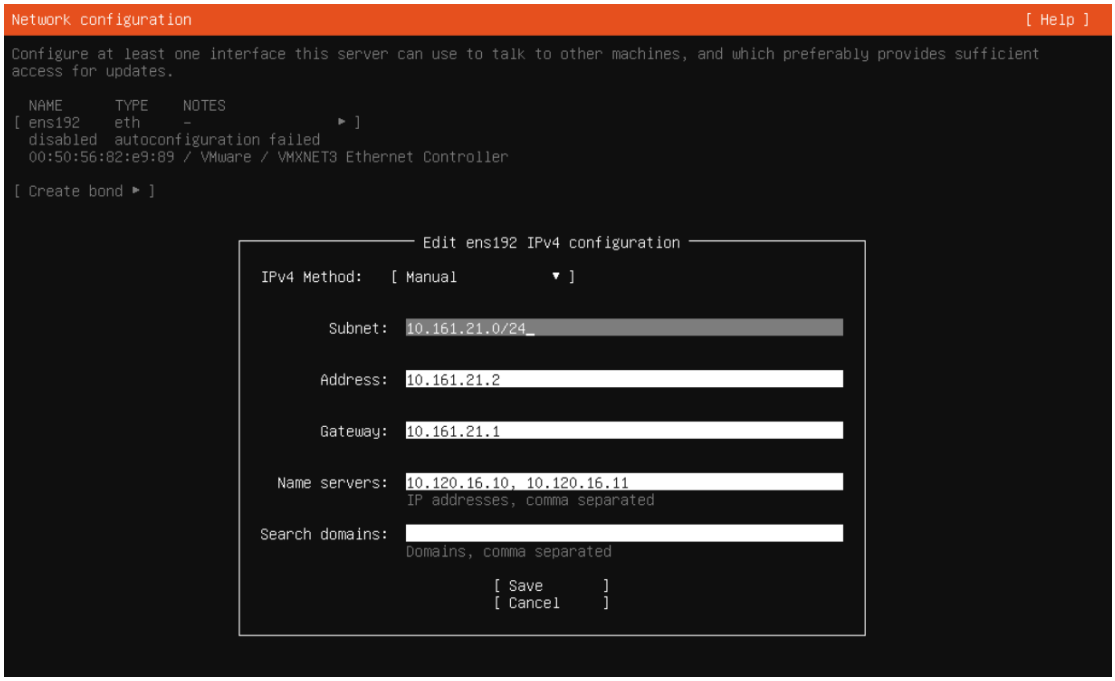
Final VM Specifications:



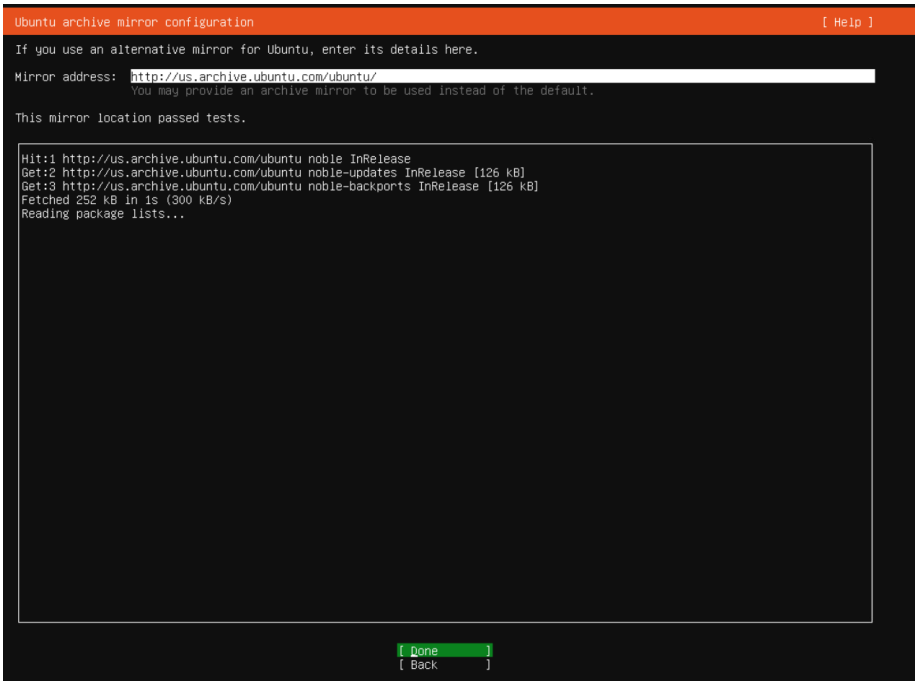
Ubuntu Server Setup

Network Configuration

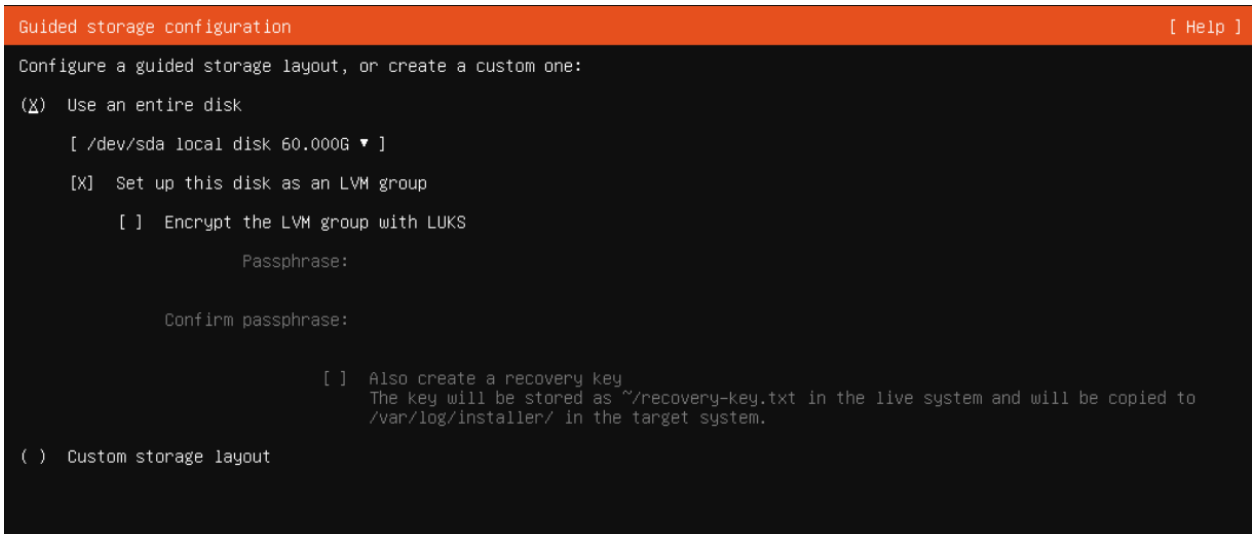
Subnet: 10.161.21.0/24 (.21 is signifying the subnet I was assigned for the semester)
Linux Server Static IP: **10.161.21.2**
Gateway: 10.161.21.1
Nameservers (DNS): UNI DNS Servers - 10.120.16.10 and 10.120.16.11

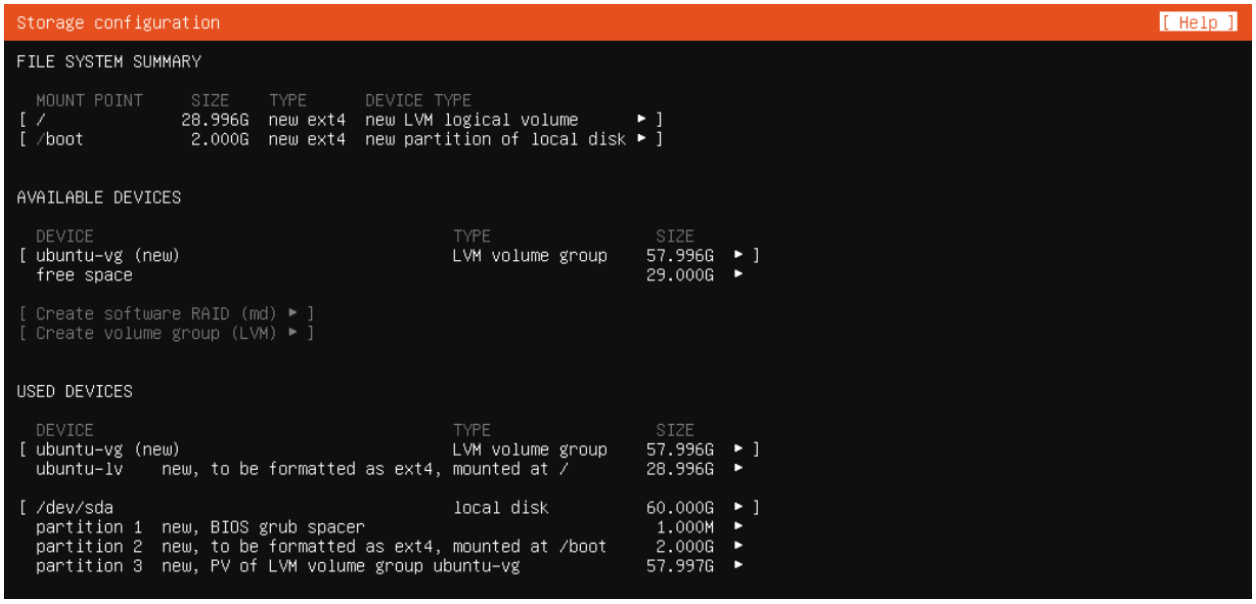


Assigned **NO** Proxy. A proxy is a “middle man” to filter network traffic coming out and/or in
Linux Mirror: US Repository Address

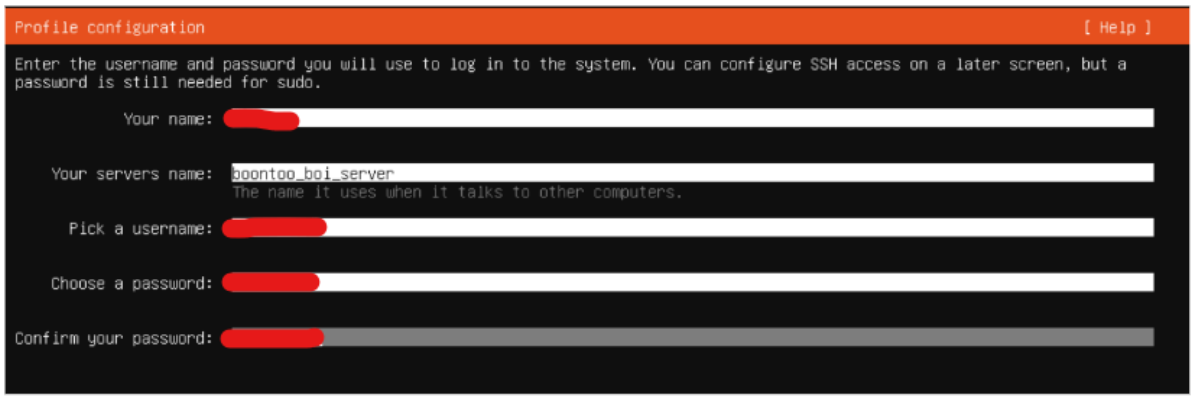


Continue **WITHOUT** Updating the Installer
Storage Configuration: Use entire disk with 1 partition

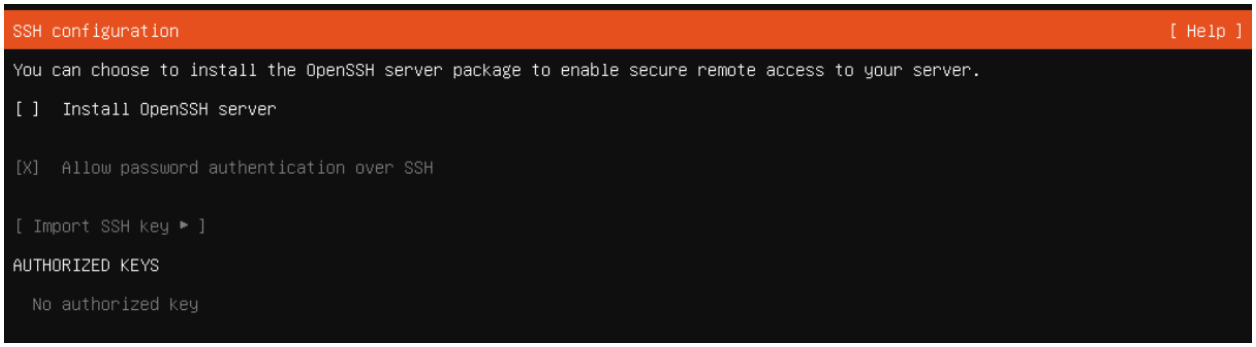




Profile Configuration: Username: **Classified**, Password: **Classified**



SSH Configuration: **NO** OpenSSH Server



Linux Server Guide

Directories

- **/home -** a user's folder that contains personal data and configuration files for programs on the machine. System programs and applications are NOT stored here.
- **/var/www** - a common place to store web content such as HTML, CSS, JavaScript, etc. The var directory is known to store "variable" (changing) data which makes sense since websites frequently update.
- **/var/log** - contains log files generated by system and user applications. The logs can include system messages, error messages, security alerts, etc. If someone was accessing my server via ssh, I would likely be able to see a log of where and when the person logged in.
- **/etc** - contains system-wide configuration files that control various aspects of the operating system and system processes. It typically includes configuration files for system services, networking, user accounts, and security settings.For example, the /etc/passwd file contains usernames and user ids while the /etc/shadow file stores hashed passwords.

Commands

- **ls -a =** the "a" flag stands for ALL so the command ls -a is "listing ALL files"
- sudo = "sudo" is short for "super user do". This is used to allow the current user to have elevated command privileges. It is commonly used to perform administrative tasks within the terminal that require higher clearance than the regular user. The sudo command prompts you for your password to confirm your identity and then allows admin privileges.
    - You can use the sudo command to update packages and software on the device within the terminal using the two commands: sudo apt update and sudo apt upgrade.

Package Management

How do you look for packages to install?

- You can search for packages using the following command: `sudo apt search <package_name>`. This command will list all packages that match the search term.
- You can view all available versions of a package with the command: `apt list --all-versions <package_name>`
- You can also see what packages have newer versions by running the command: `sudo apt update` and then: `sudo apt upgrade` to upgrade all installed packages to their newer versions.

How do you install a package?

- You can install a package using the command: `sudo apt install <package_name>`. This command will install the specified package and its dependencies.

How do you see what packages are already installed?

- You can check for installed packages using the command: `apt list –installed`. This command will list all of the system's **installed** packages.

How do you remove a package?

- You can remove a package with the command: `sudo apt remove <package_name>`. This command will remove the specified package but leave the configuration files. You can use the "purge" keyword to recursively delete package files and its configurations with the command: `sudo apt purge <package_name>`.
- After removing or purging a package, you can clean up any unused dependencies with the command: `sudo apt autoremove`.

Services

- You can list all running services with the command: `systemctl list-units --type=service --state=running` or `service –status-all`.
- Start a service: `sudo systemctl start <service_name>`
- Stop a service: `sudo systemctl stop <service_name>`
- Restart a service: `sudo systemctl restart <service_name>`
- Service status: `systemctl status <service_name>`

Cron Jobs

Cron Jobs are automated scripts that run on specific time intervals. You can find cron jobs in a variety of different places within the /etc directory:

- **/etc/cron.hourly** = hourly scripts
- **/etc/cron.daily** = daily scripts
- **/etc/cron.weekly** = weekly scripts
- **/etc/cron.monthly** = monthly scripts
- **/etc/crontab** = the system wide cron table where jobs are defined and scheduled, can be accessed by super user -> sudo
- **/etc/cron.d** = application specific cron jobs and system maintenance tasks
- **/var/spool/cron/crontabs/<username>** = user specific cron jobs are located here (you can create and edit cron jobs with the command: `crontab -e`)

Adding Cron Jobs

- You can add cron jobs to the current user using the crontab utility: `crontab -e` or the root user using the command: `sudo crontab -e`
- You then can add a cron job with a path to your script in this format: * * * * * **/path/to/your/script.sh**
- You can also add a cron job file to the **/etc/cron.d** directory by using the command: `sudo nano /etc/cron.d/my_cron_job`. You then add the path to your script with the same format as above!

Crontab Commands:

- `crontab -e` = create crontab file for current user
- `crontab -l` = list cron jobs for the current user (my user has none hence the comments only)
- `crontab -r` = remove crontab file

```
crontab: installing new crontab
onelessone@boontooboiserver:/etc/cron.d$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
#
```

User Management

Adding Users

1. `sudo useradd <new_username>` = adds new user
2. `sudo passwd <new_username>` = set a password for new user
3. `sudo useradd -m <new_username>` = adds a new user with a home directory (-m flag)

Removing Users

1. `sudo userdel <username>` = deletes a user (does not remove their home directory)
2. `sudo userdel -r <username>` = deletes a user and their home directory (-r flag)

File Permissions

First character indicates the **type of file**:

- d = directory
- - = regular file
- l = symbolic link

Next nine characters represent the permissions for the owner, group, and others:

Owner Permissions (first three characters): Example -> rwx

- r = read permission
- w = write permission
- x = execute permission

Group Permissions (next three characters): Example -> r-x

- r = read permission
- - = NO write permission
- x = execute permission

Others Permissions (last three characters): Example rw-

- r = read permission
- w = write permission
- - = NO execute permission

Changing Permissions

You can change the permissions by using the chmod command.

Numeric Method

Ex: chmod 755 <filename>

Each permission is represented by a number:

- r (read) = 4
- w (write) = 2
- x (execute) = 1

755 means:

- Owner: 7 (4+2+1 = rwx)
- Group: 5 (4+1 = r-x)
- Others: 5 (4+1 = r-x)

Symbolic Method

Ex: chmod u+rwx,g+rx,o+rx <filename>

- u (user/owner), g (group), o (others)
- + (add permission), - (remove permission)

Ex: What does “drwxr-xr-x” mean?

d = directory.
rwx = the owner has read, write, and execute permissions
r-x = the group can read and execute but not write
r-x = others can read and execute but not write

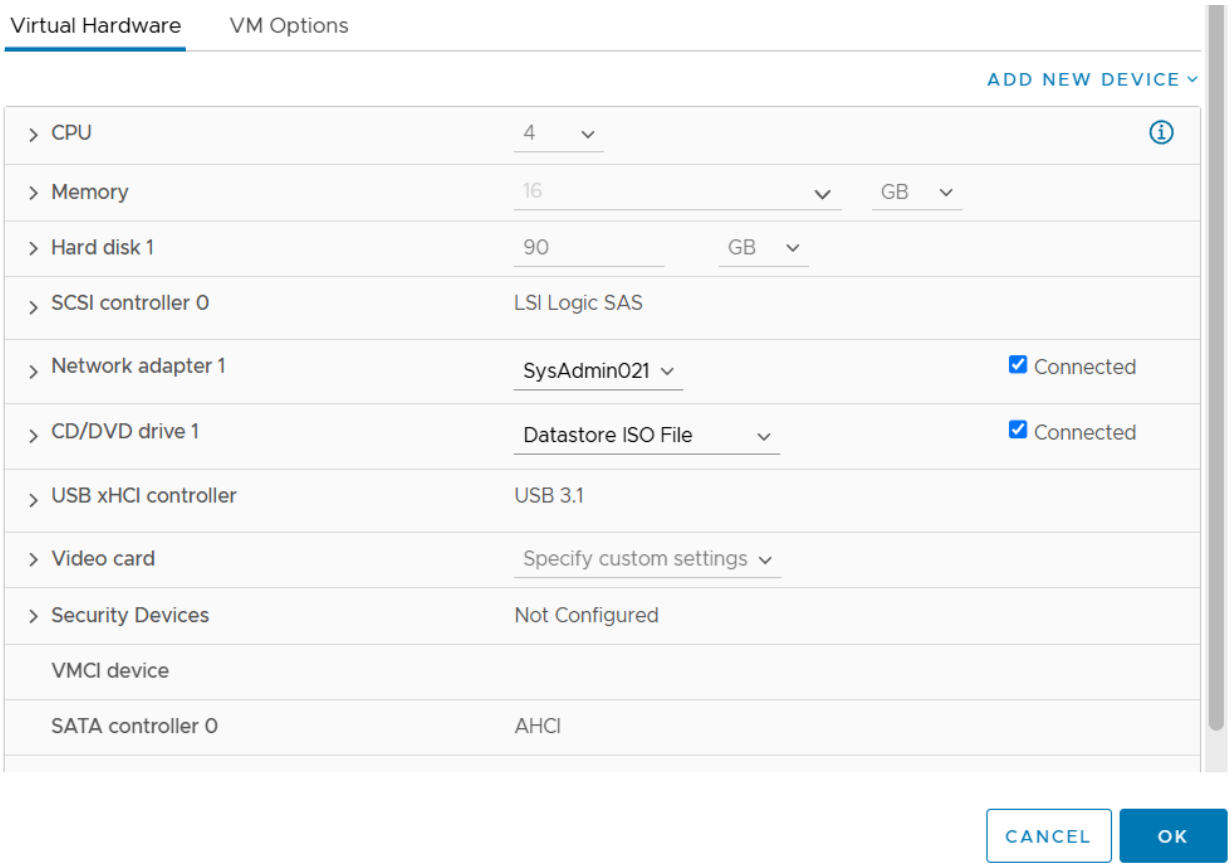
Project 2: Windows Server

Below contains steps outlining the configuration of a **Windows 2022 Server** that will be used to initially configure pfSense through the admin portal in a browser and serve as an Active Directory Domain Controller. The Windows Server will start in the WAN under the Static IP Address: **10.161.21.3**.

VM Configuration

Guest OS: Microsoft Windows Server 2022
Hardware: 4 CPUs, 16 GB RAM, Thin Provision, Use Datastore ISO file
ISO File: **Windows Server 2022**

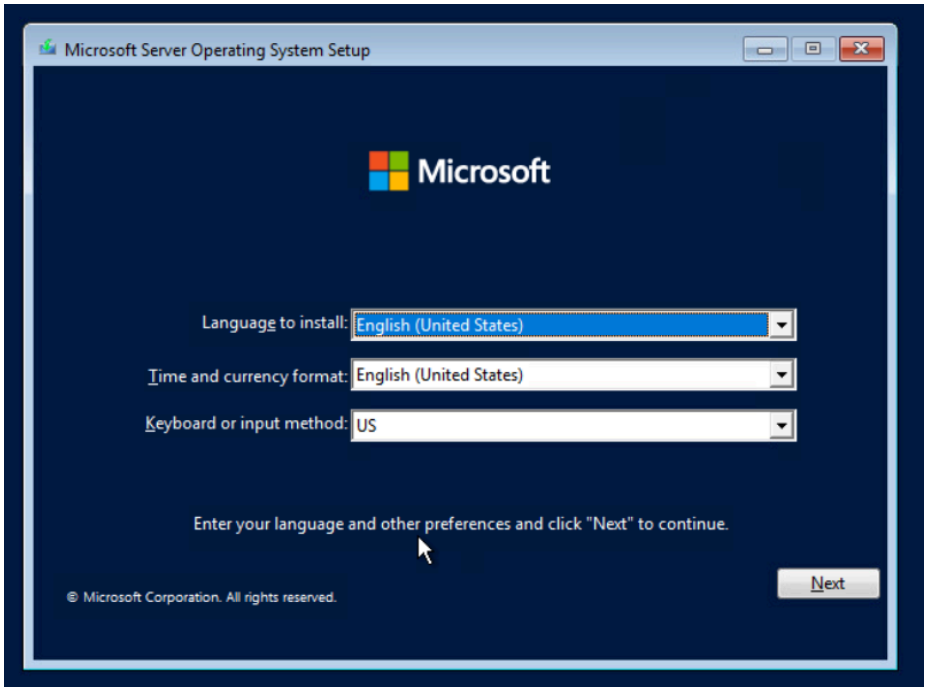
Final Specifications



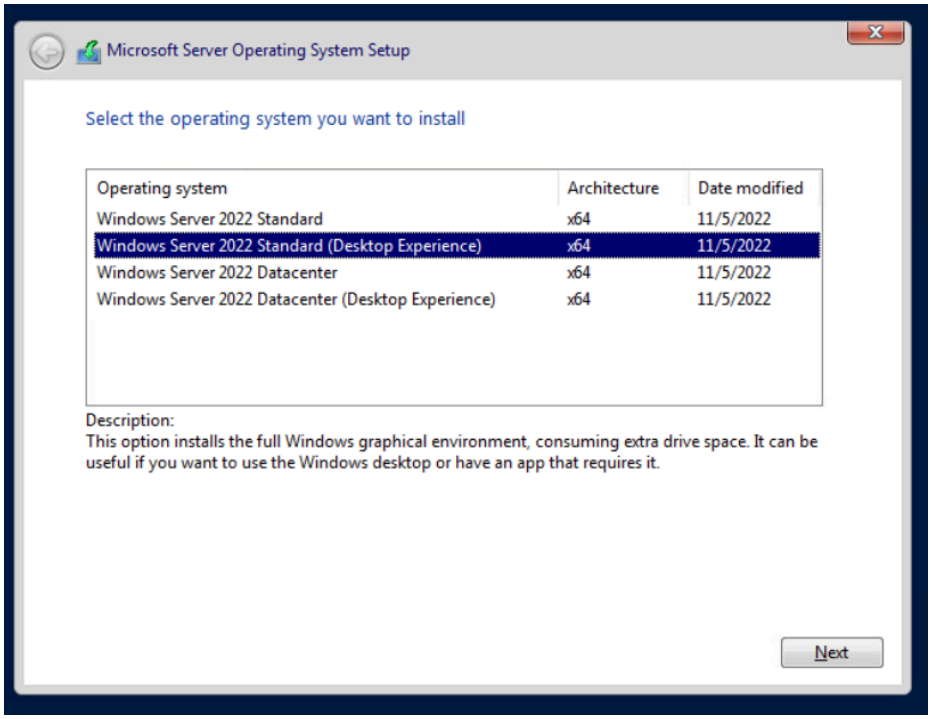
Windows Server Setup

Windows Installer

Select Language and Keyboard and then click “Next”.

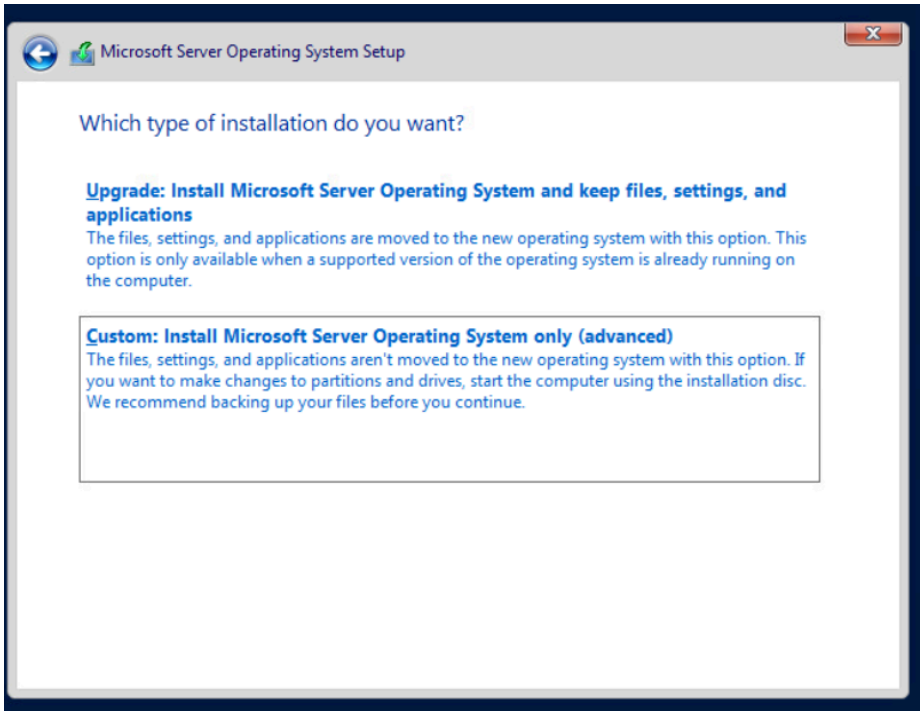


OS Choice: Windows Server 2022 (Desktop Experience)

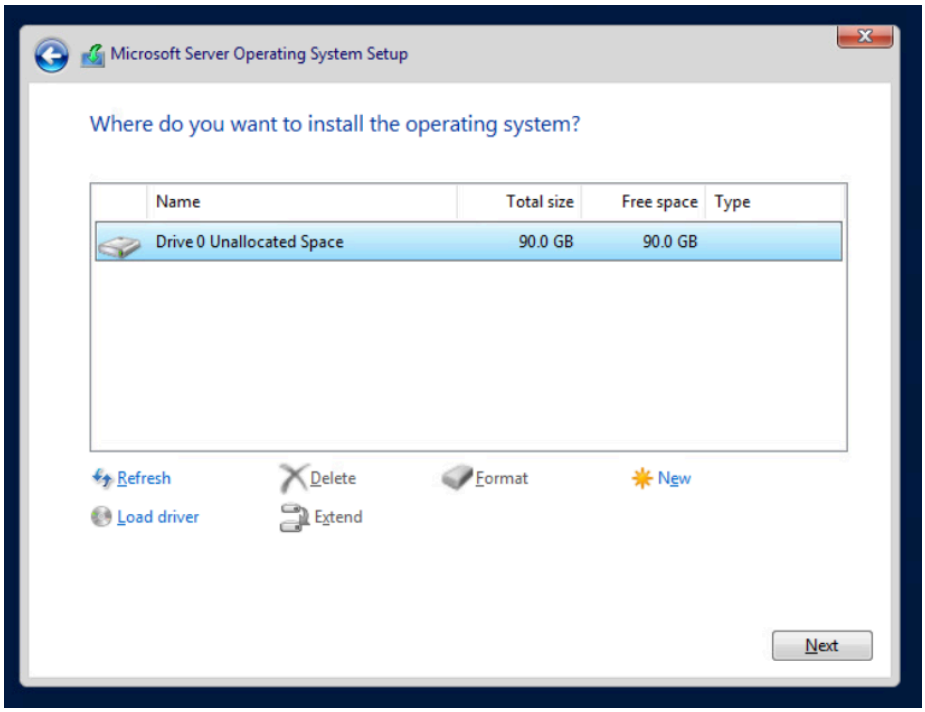


Accept Terms and Conditions ✅

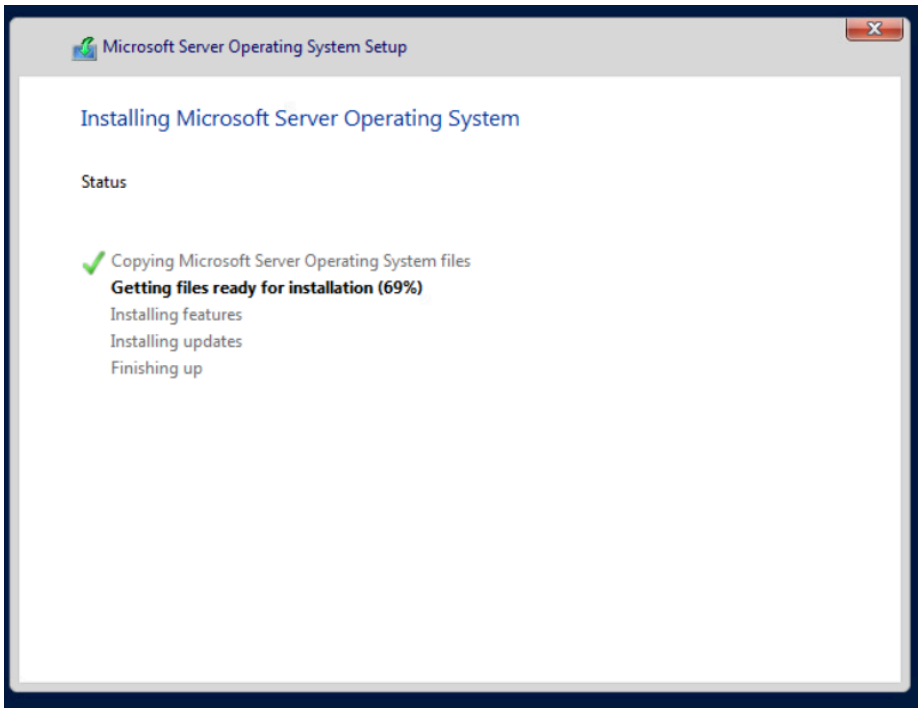
Designate a **Custom Install**



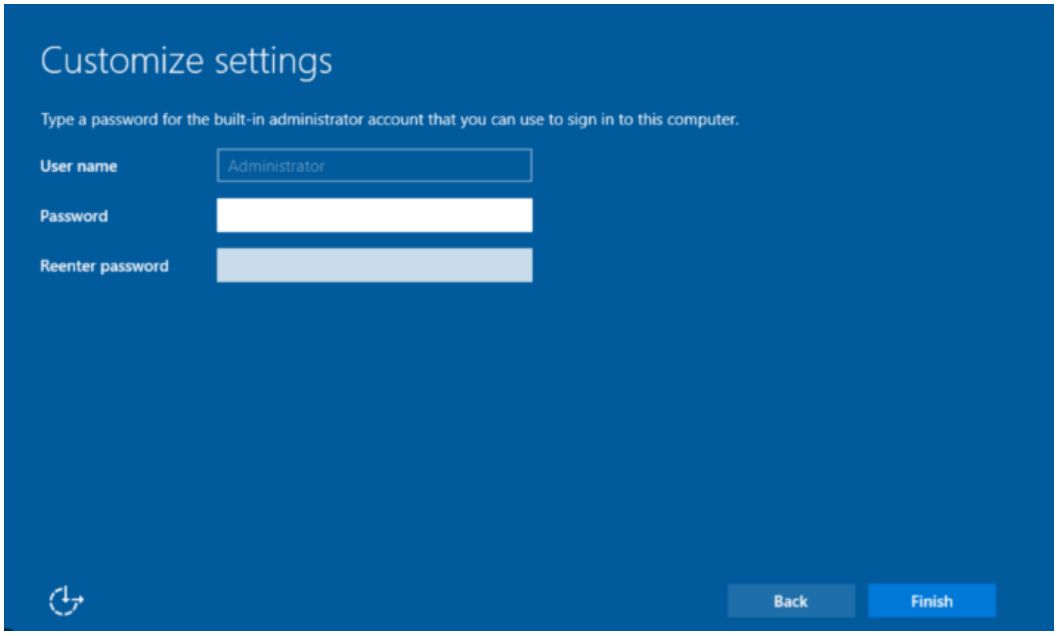
Do not edit the drive. Click **NEXT** to use the full partition.



Sit Tight and Watch the Installation Screen. If you stare too long you might become a Grandpa👴!



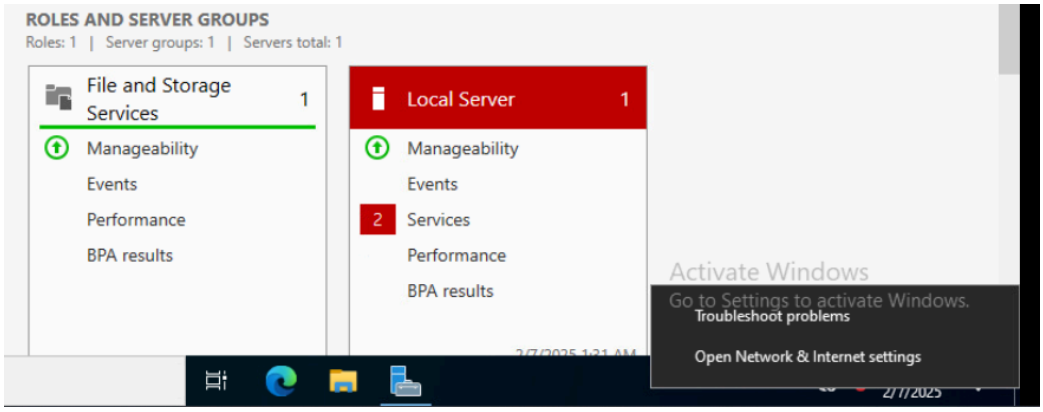
Set Administrator Password



You will then be prompted to **restart** the device to solidify the installation.

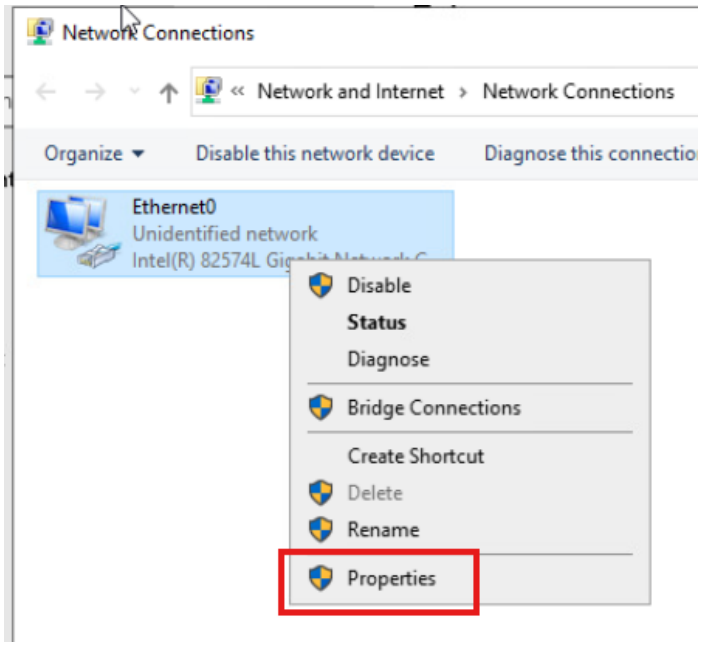
Windows Server Configuration

Login to Windows after restart and setup Network Settings by right clicking the “Globe” on the bottom right and selecting “Open Network and Internet settings”

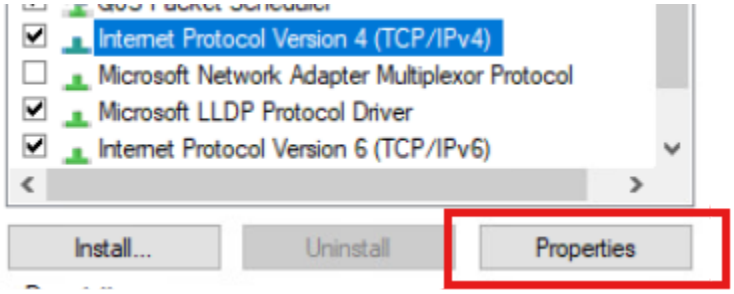


Navigate to Ethernet and “Change adapter options”

Right click the Ethernet icon and select “Properties”

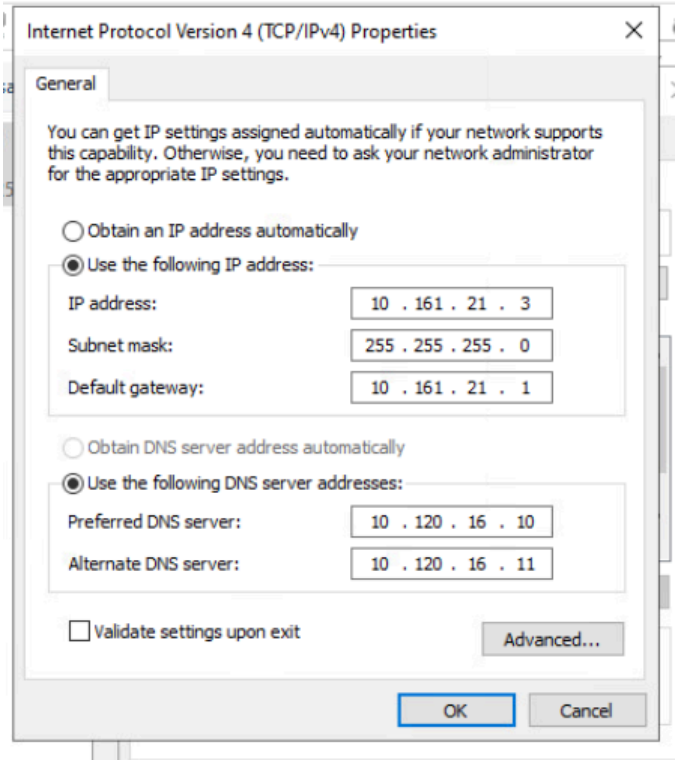


Navigate to Internet Protocol Version 4 (IPv4) and select “Properties”

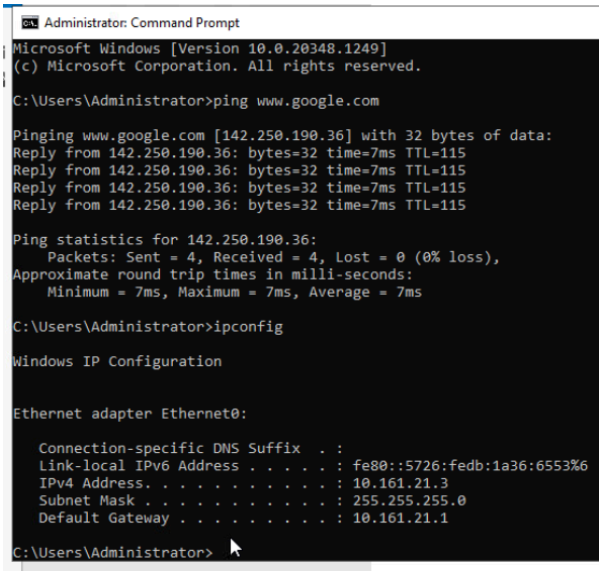


Network Configuration

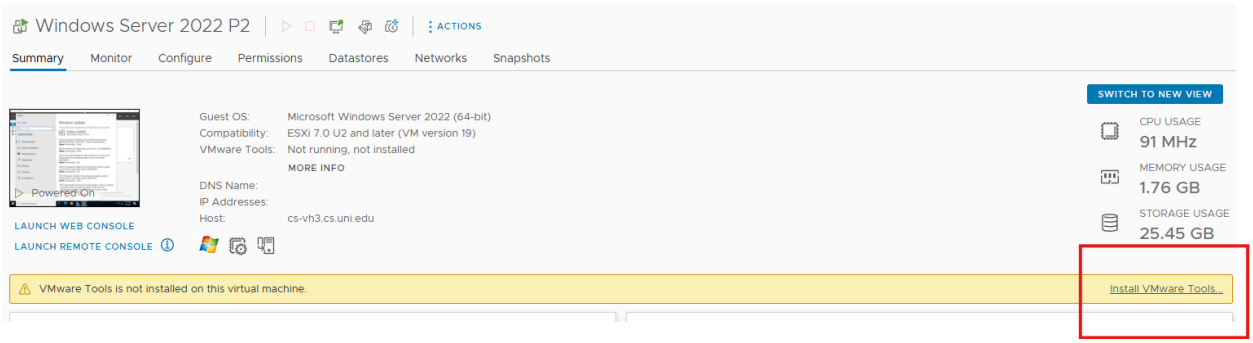
Subnet: 10.161.21.0/24 (.21 is signifying the subnet I was assigned for the semester)
Windows Server Static IP: **10.161.21.3**
Default Gateway: 10.161.21.1
DNS Servers: UNI DNS Servers - 10.120.16.10 and 10.120.16.11



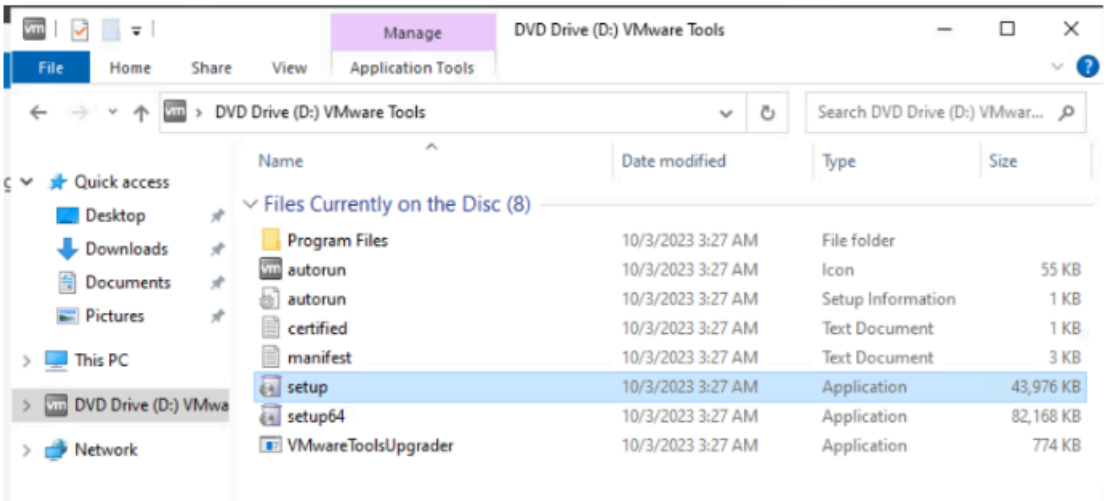
Go to terminal and ping www.google.com to verify network connectivity



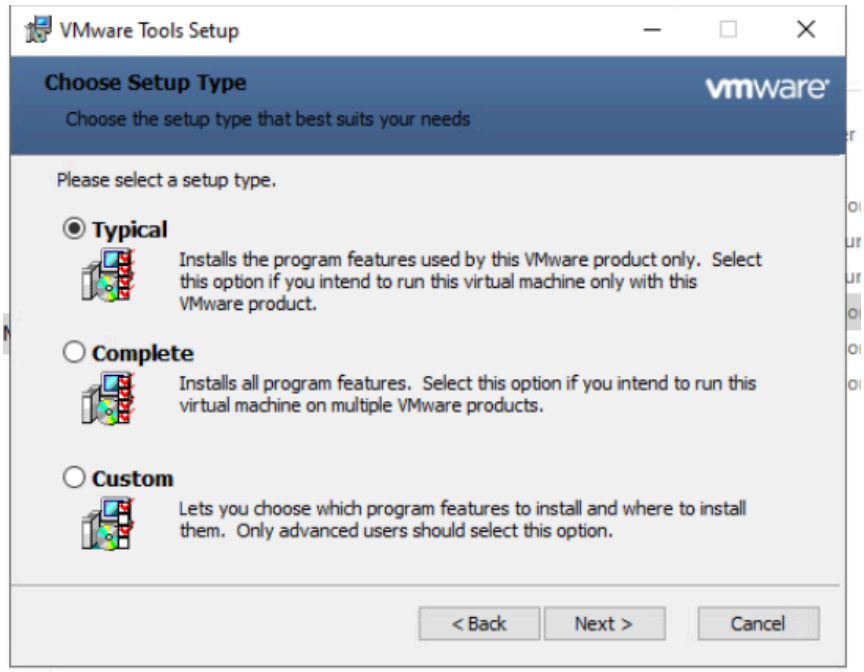
Go to Vsphere Esxi and Install “VMware Tools”



Go back to the Windows Server 2022 VM and navigate to File Explorer to open the newly mounted DVD Drive



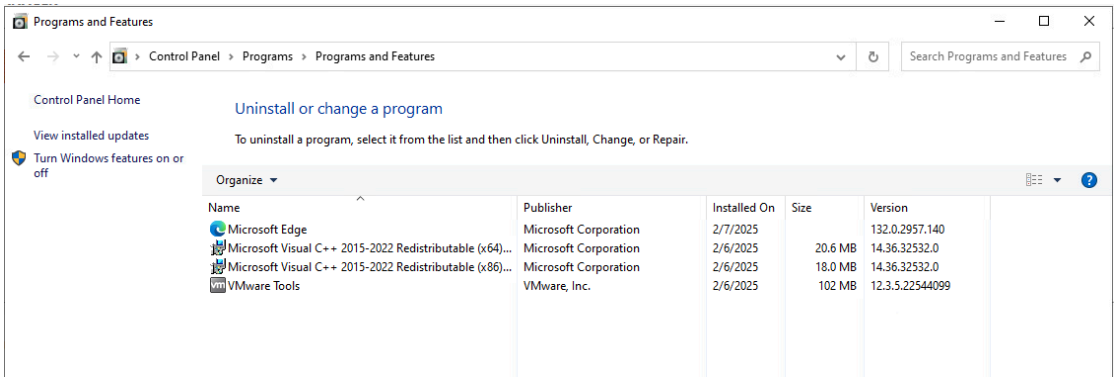
VMWare Tools Install: Typical
Once install is complete, **restart** the System



Windows Server Guide

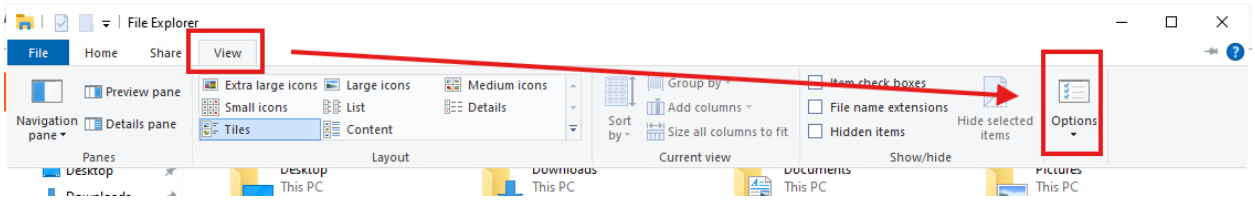
Manage Installed Programs

1. You can access the “Control Panel” program through the Start Menu
2. Navigate to “Programs”
3. Now you can see all of the installed programs and manage them (uninstall, repair, change, etc.)



Show Hidden Files and Directories

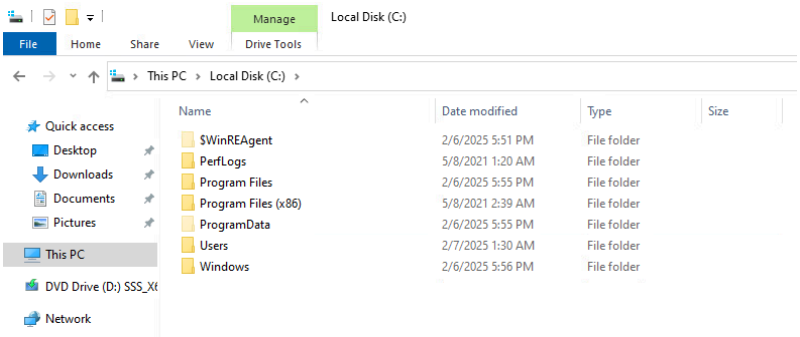
Navigate to File Explorer and select “View” on the top bar. After selecting “View”, select the “Options” module



A popup will appear. Navigate to “View” in the top bar and then select “Show hidden files, folders, and drives”.

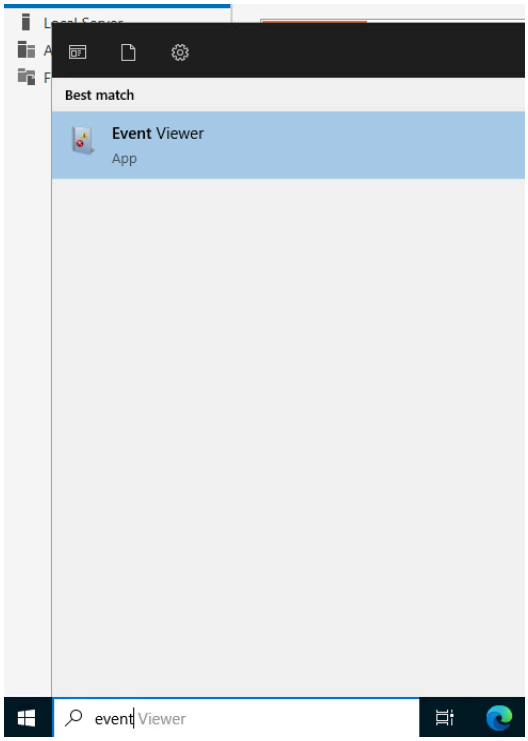
After selection, click “Apply”. Hidden files and folders will now be visible in the File Explorer GUI. **Hidden files and folders will generally appear with less opacity** (transparent) while files that are always visible still appear with full opacity.

Example: **ProgramData Folde**r (now visible)

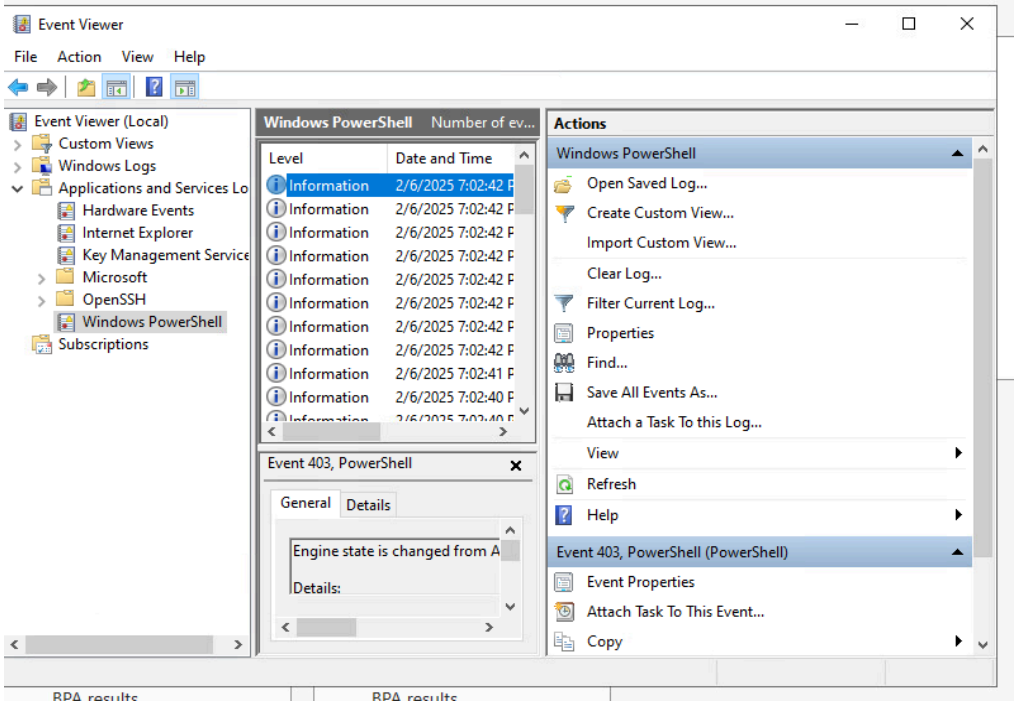


Windows System Logs

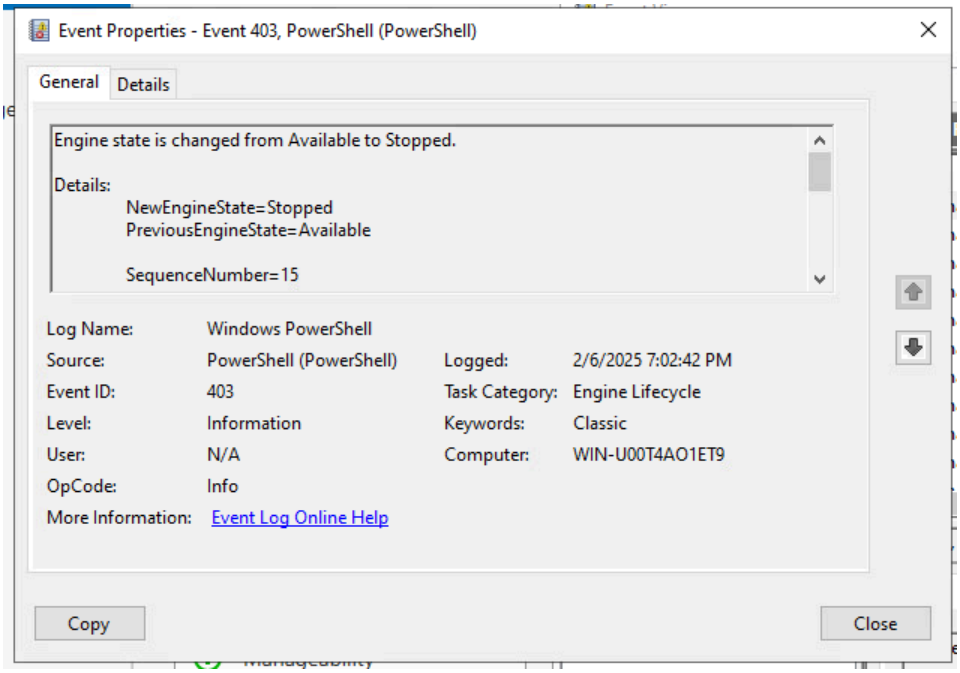
Launch “Event Viewer” from the Start Menu



Once in the application, you can use the side menu to select from a variety of categories and view their logs



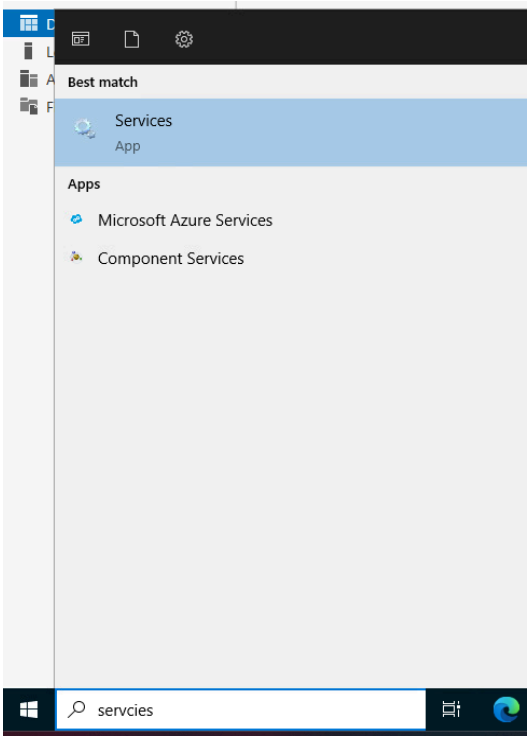
You can click on an individual log (event) and see its exact details

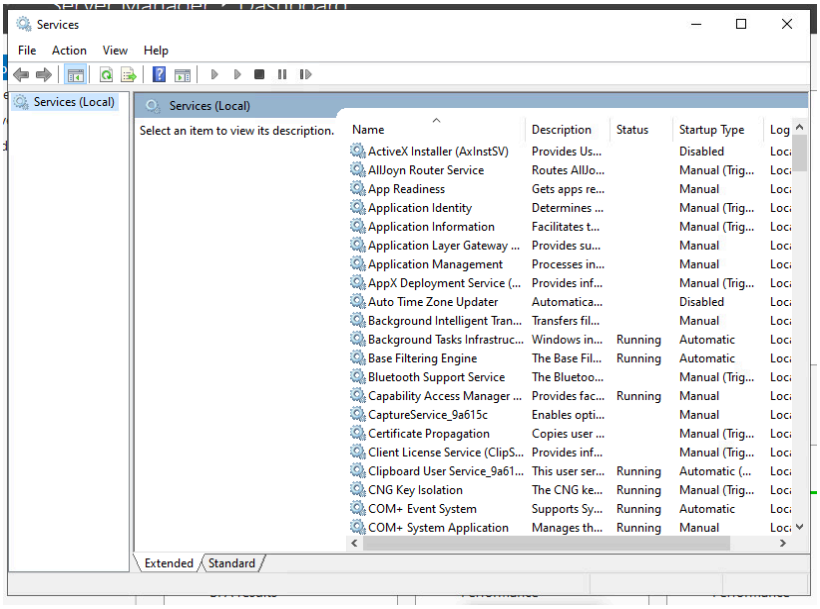


Windows Services Manager

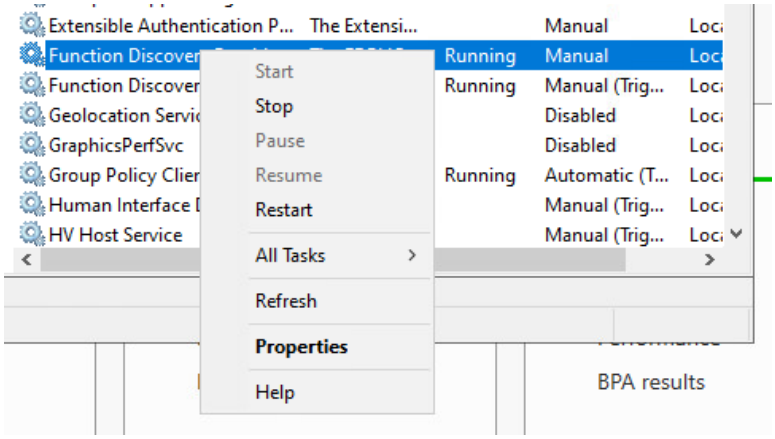
The Windows Services Manager is a tool that allows you to manage the services running on your Windows operating system. Services are background processes that perform various functions, such as networking, security, and system maintenance.

Launch “Services” from the Start Menu to access the Services Manager.

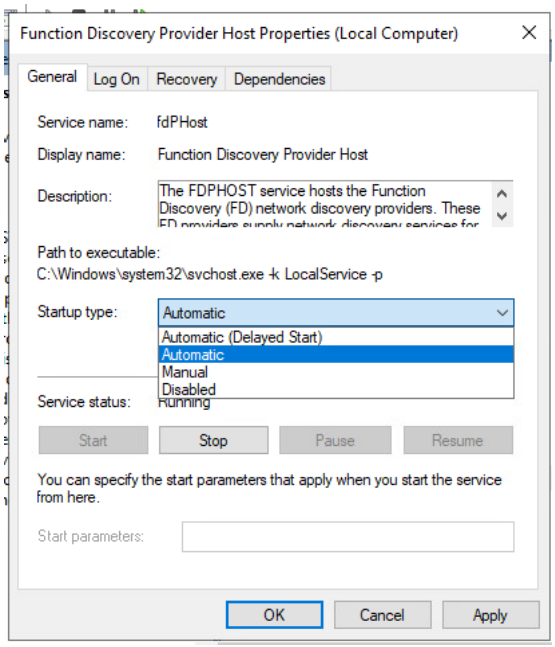




You can **start**/**stop**/**restart** a service in the Services Manager by selecting or right clicking the desired service.

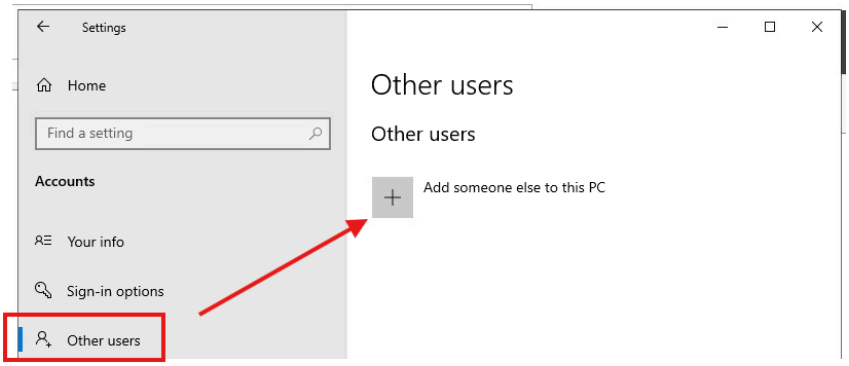


You can configure a service to start automatically by navigating to "Properties" after right clicking. You then can customize the Startup type to "Automatic".

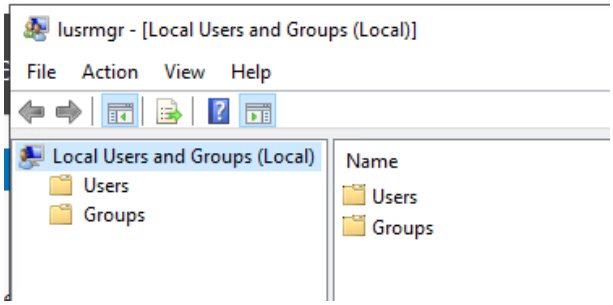


Add/Delete Local Users

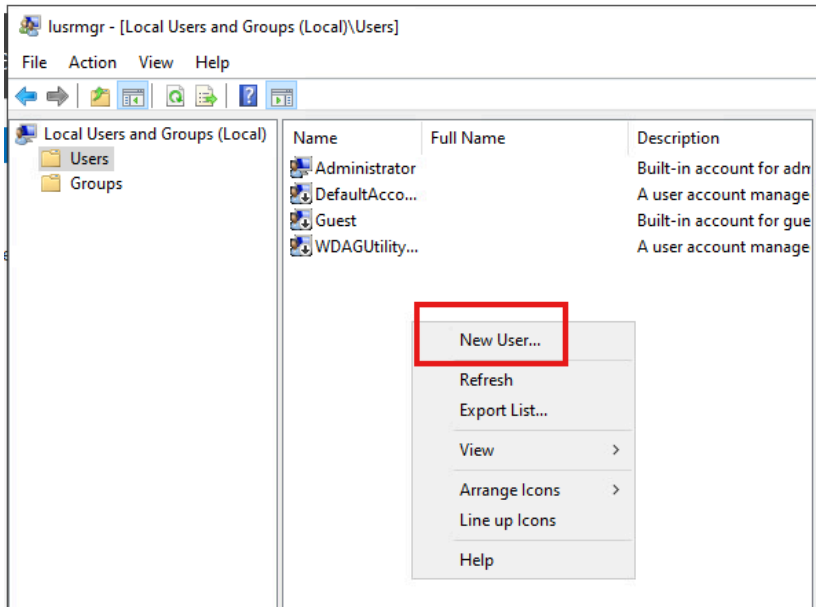
Navigate to “Settings” and then the “Accounts” tab. Once in Accounts, select “Other Users”.



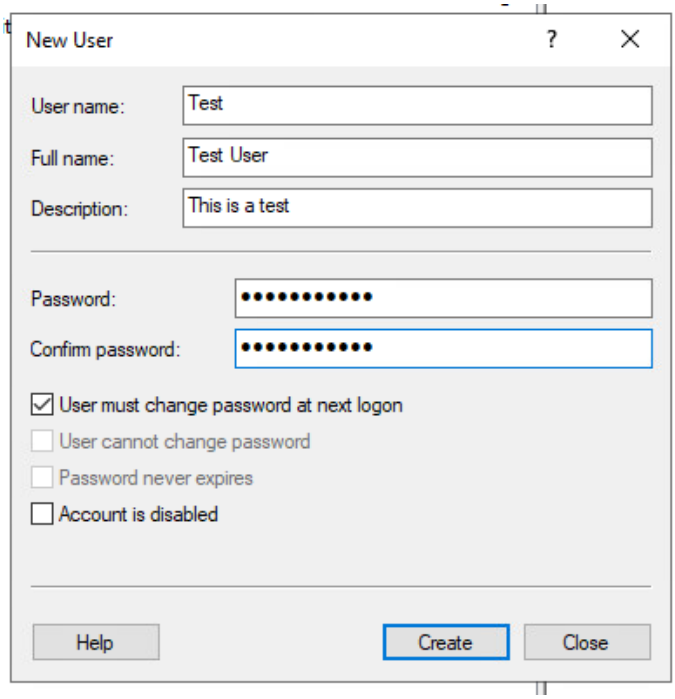
After selecting the **“+”**, a menu should appear with a “Users” and “Groups” folder in the left bar.



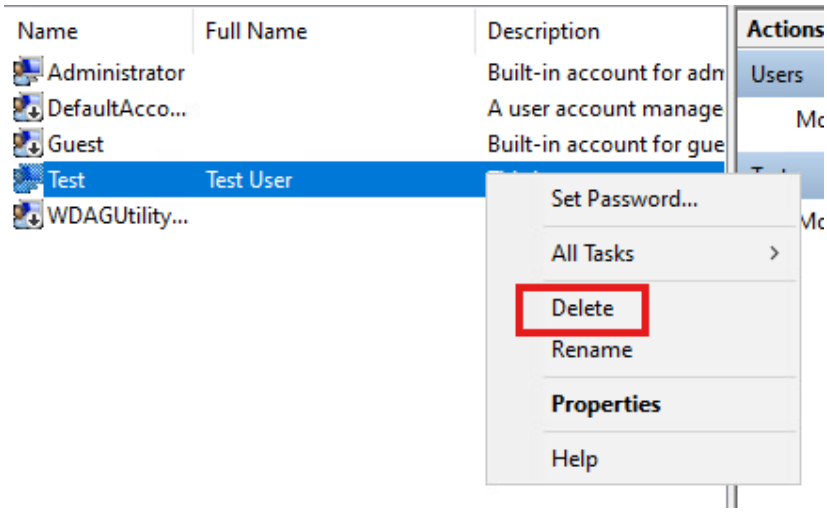
Click on the “Users” folder and then right click to bring up a popup menu to add a “New User”.



You will be prompted to enter information about the new user. Once the user is created it will show up in the Users directory.



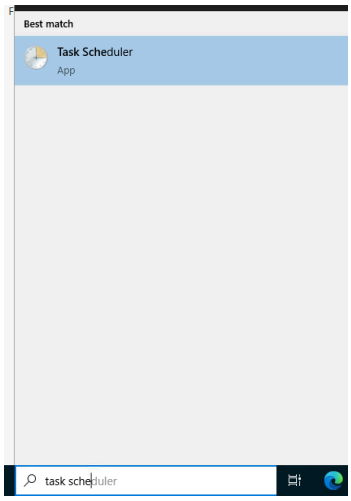
You can delete a user by right clicking an existing user and selecting “Delete”.



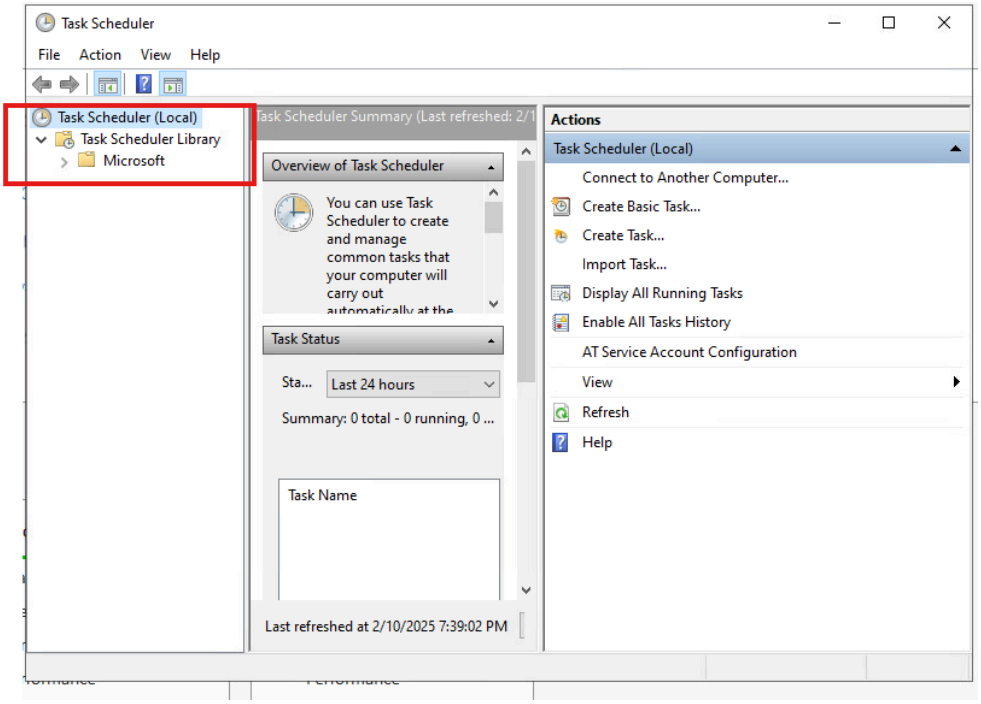
Task Scheduler

Windows has a built-in tool called Task Scheduler. It functions similarly to a crontab where you can put when and how often you want a script to execute.

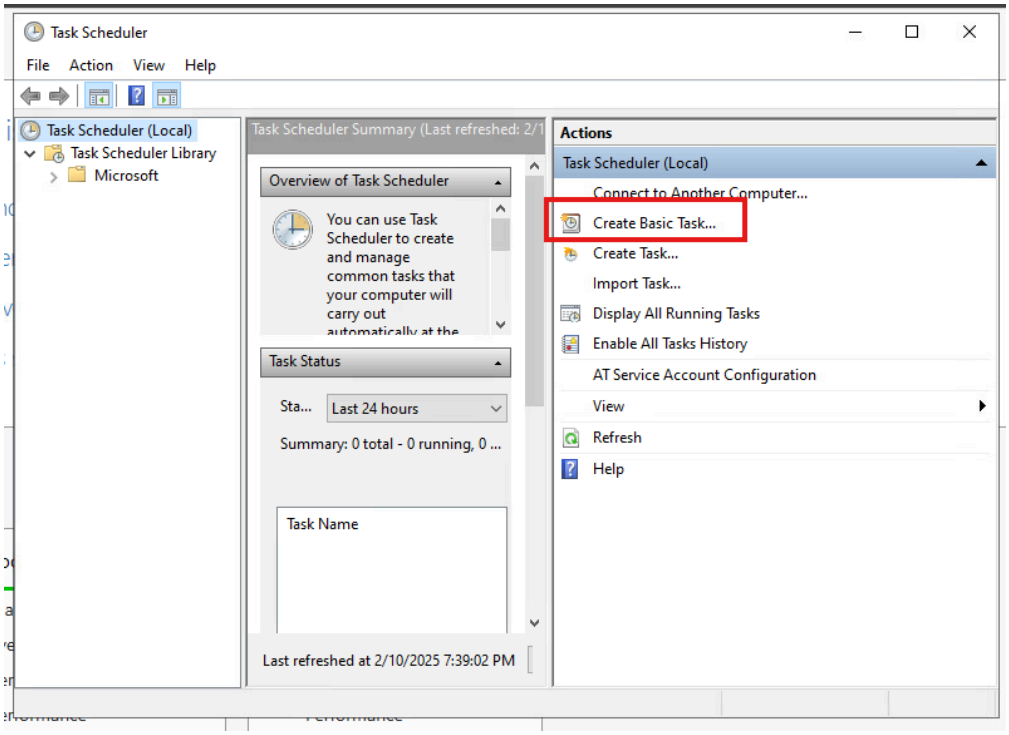
Launch “Task Scheduler” from the Start Menu



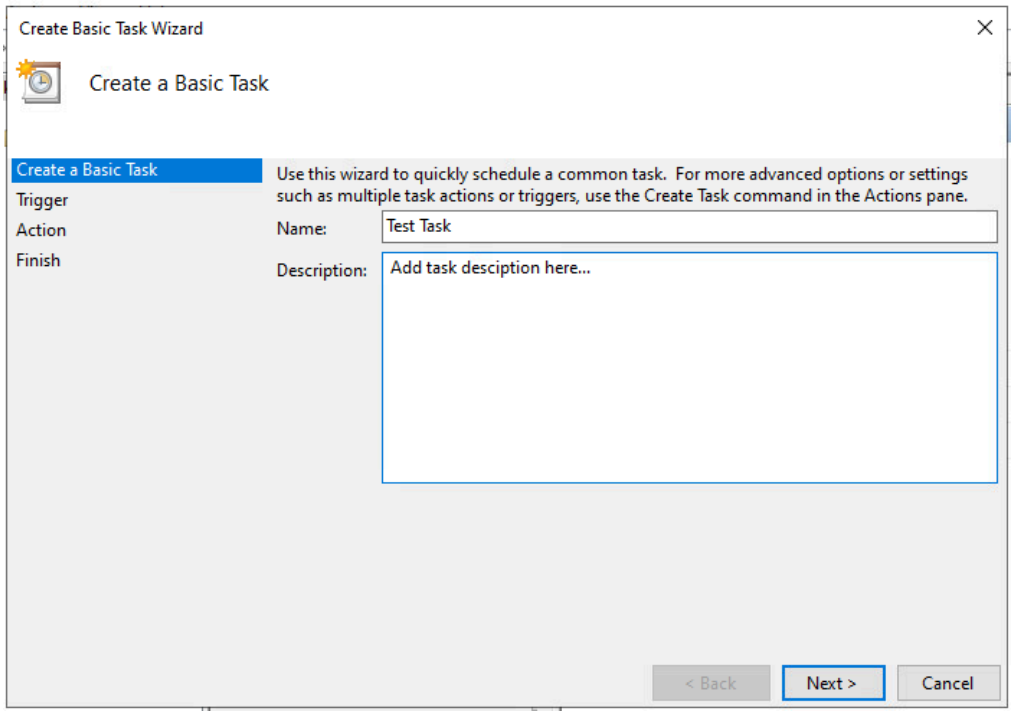
Use the left side menu to see existing Windows Tasks



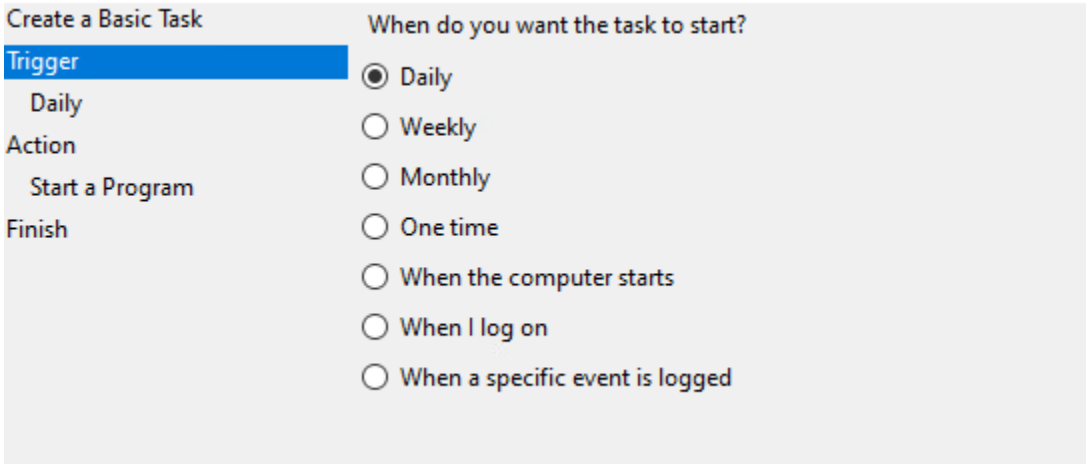
You can create a new task on the right side menu with “Create Basic Task”



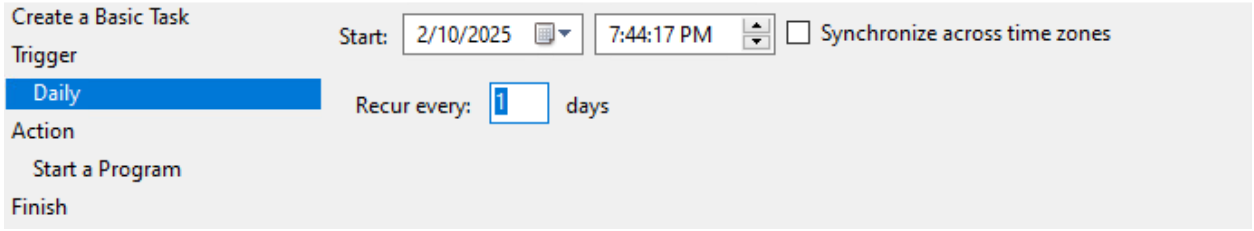
Add “Task Name” and “Description”



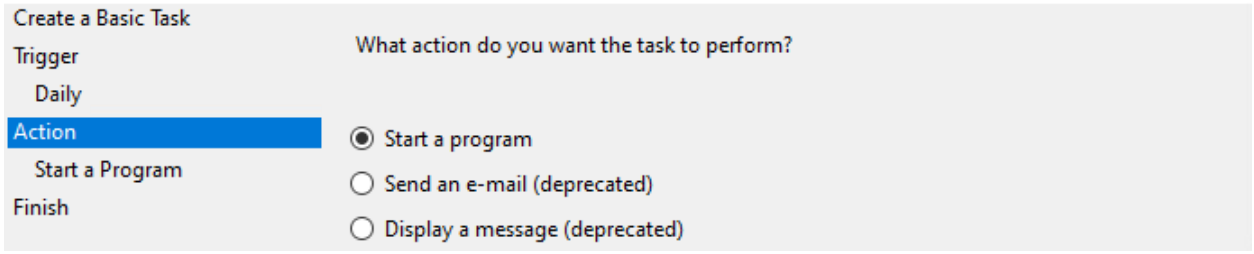
Add a task “Trigger”



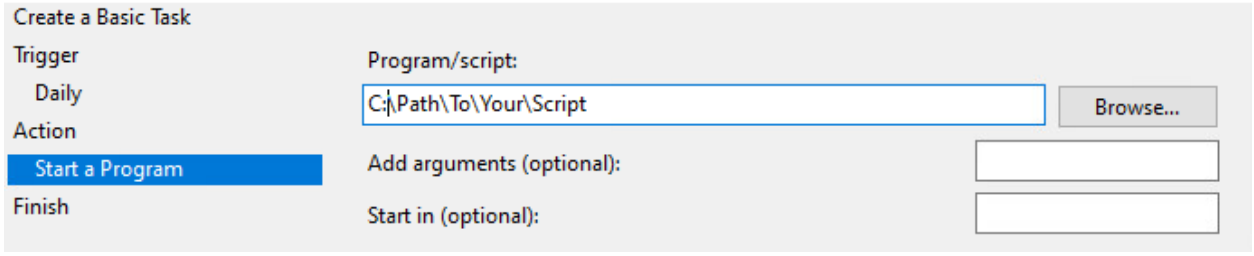
Select when you would like the task to **repeat**



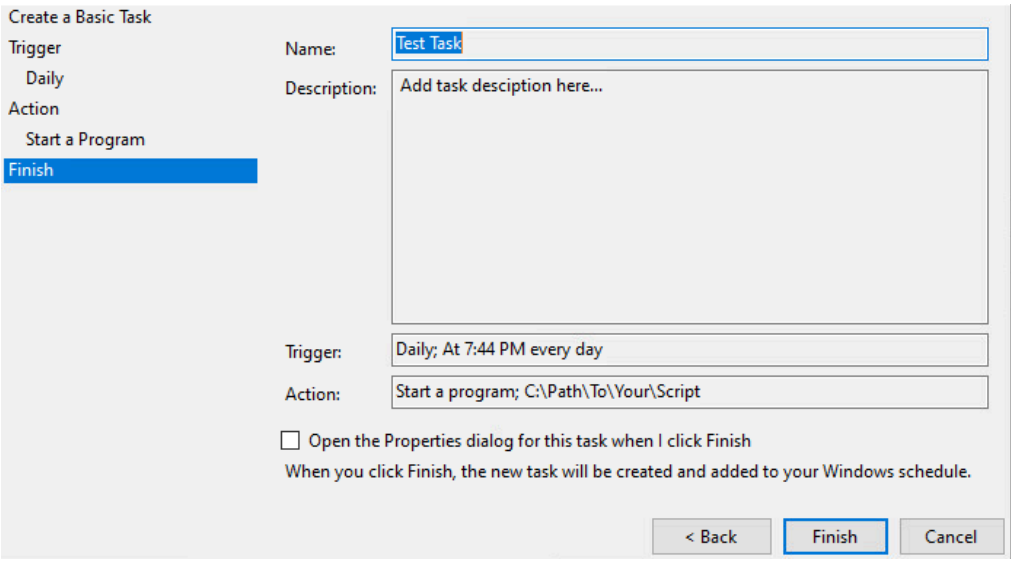
Select task Action



Configure the script you would like to execute

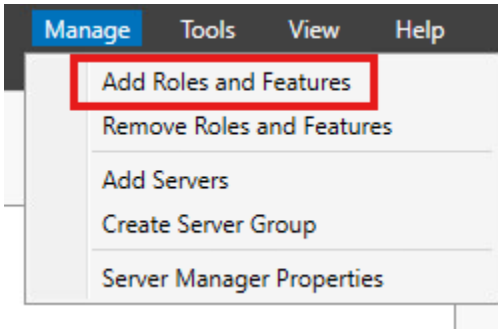


Confirm your new Task

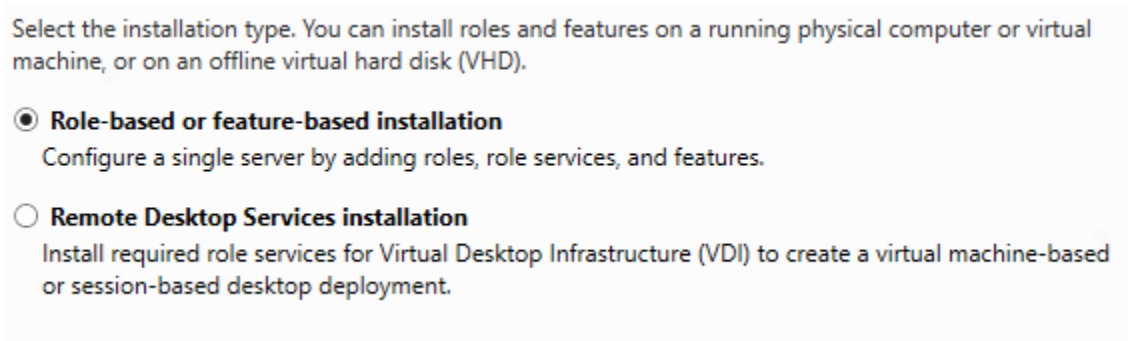


Server Roles

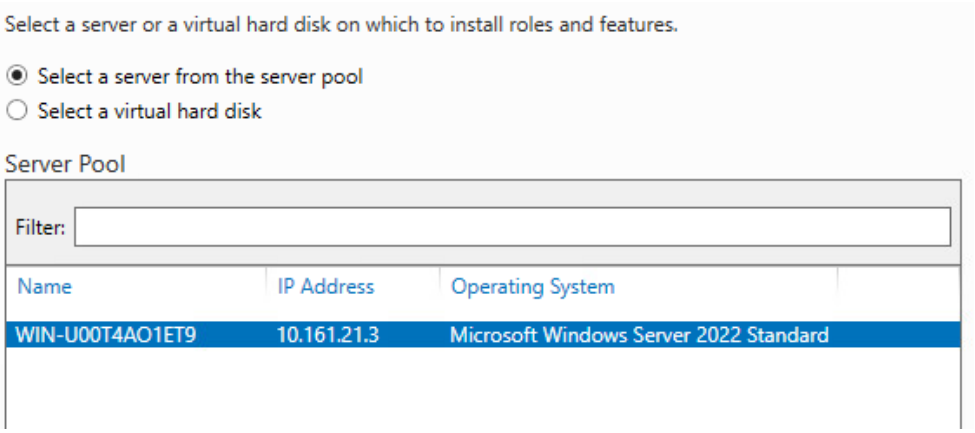
Open Windows Server Manager and go to the top right and select the drop down menu named "Manage". Then choose "Add Roles and Features".



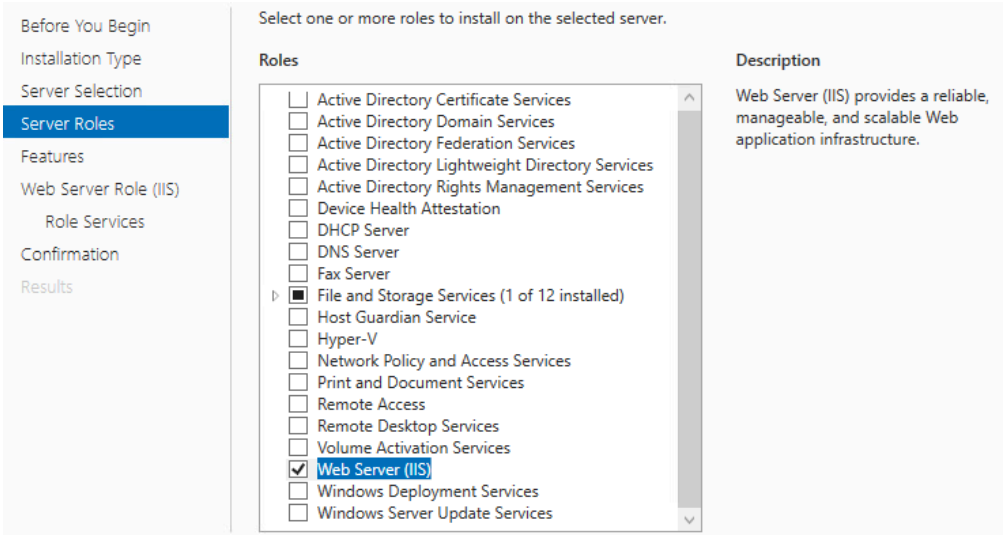
Set Installation Type to "Role-based or feature-based installation"



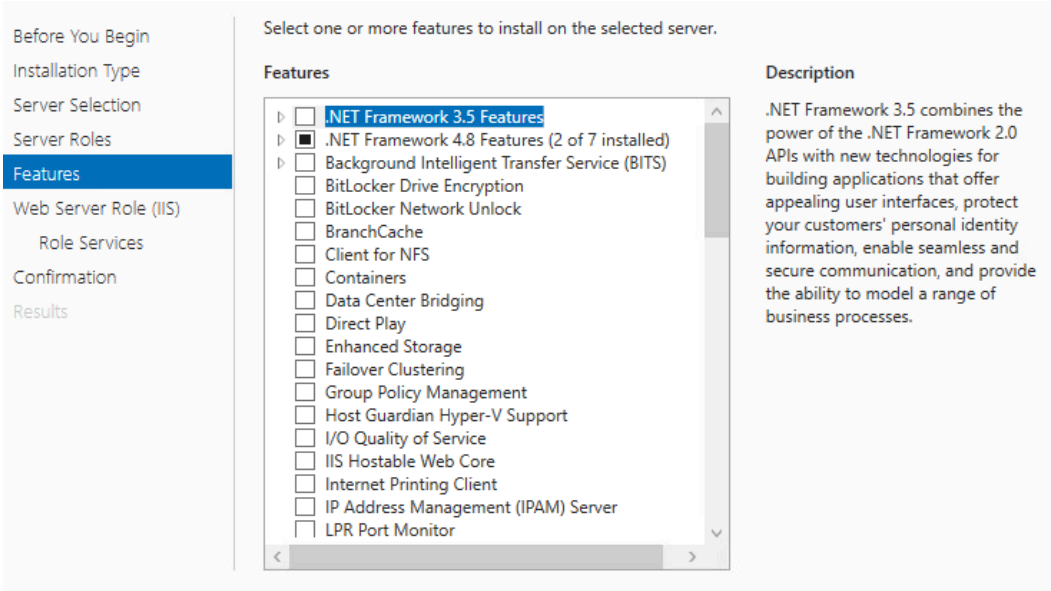
Select the desired server



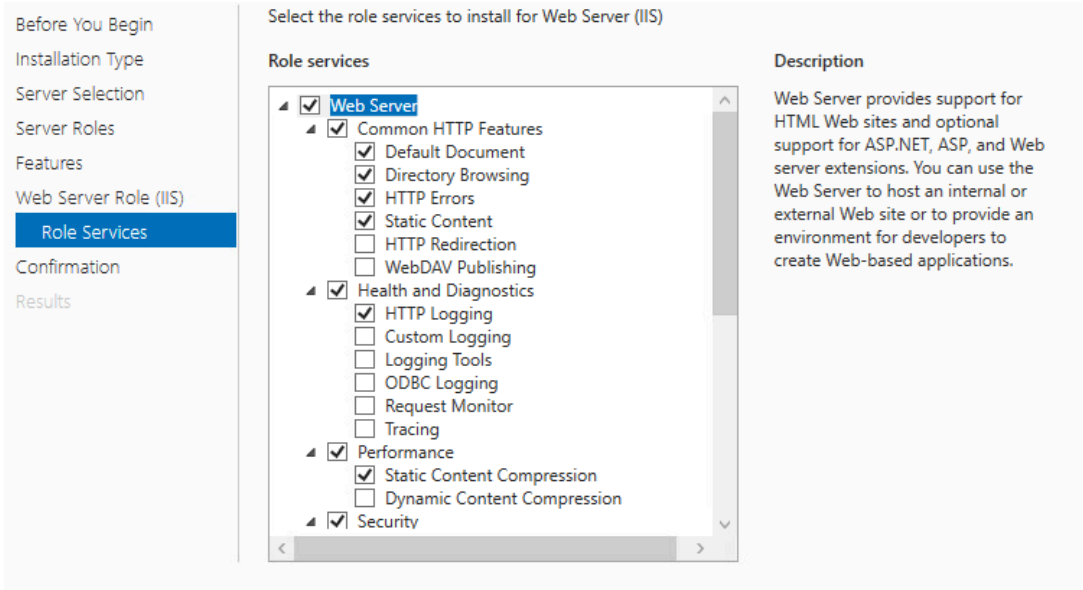
Designate **Server Roles**. Ex: Web Server (IIS)



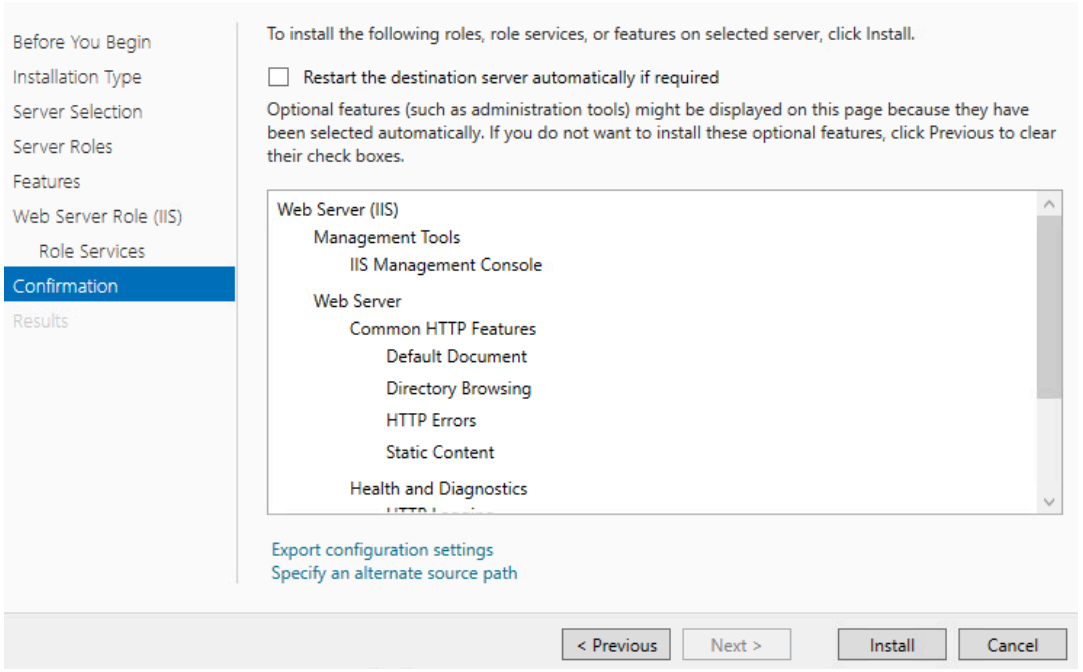
Select desired Features



Configure Role Services

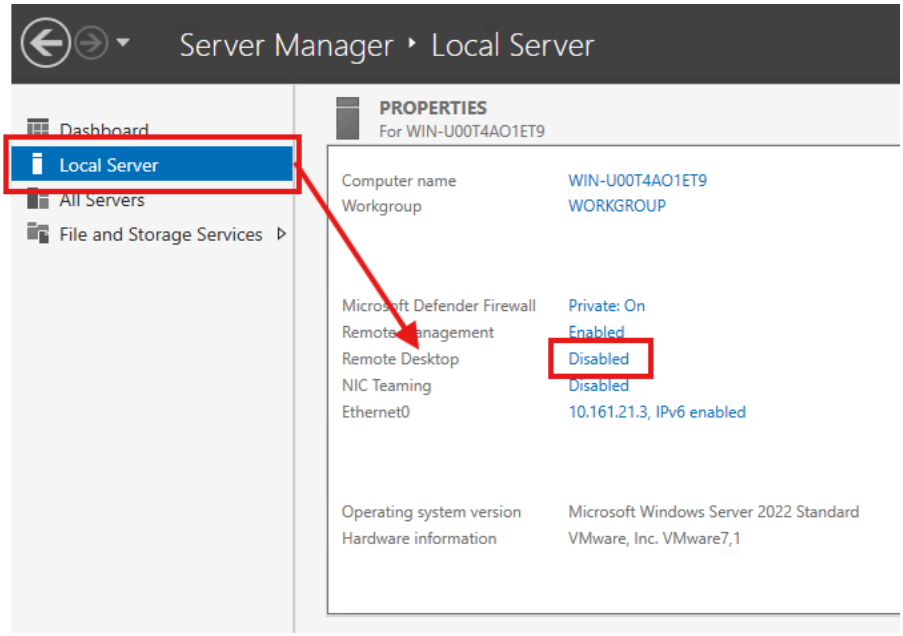


Confirm and **Install**



Remote Desktop

You can track Remote Desktop (RDP) status by going to “Local Server” in the left side menu.



Project 3: pfSense Firewall

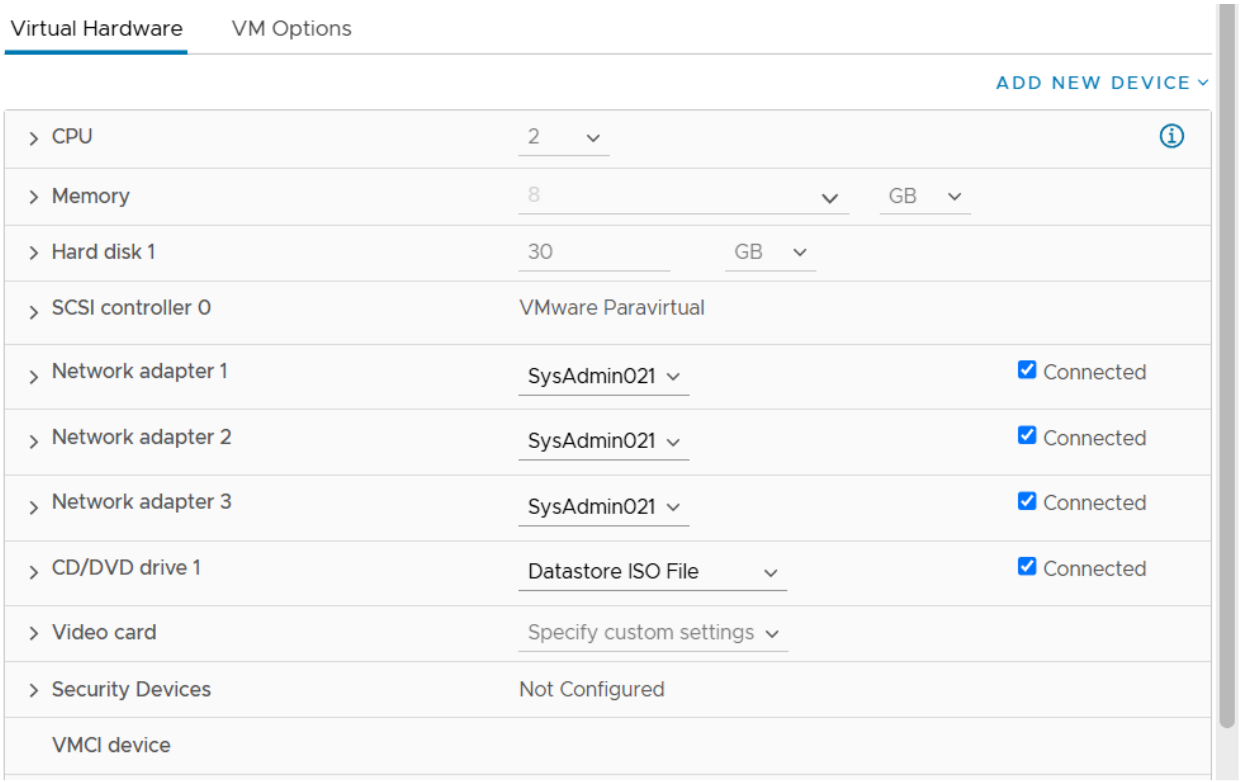
Below contains steps outlining the configuration of a **pfSense firewall** for my network cluster. It will be hosted in the WAN and allow me to configure two internal networks: **LAN** and **DMZ**. A DHCP server will be hosted on the LAN for “company workstations” and the DMZ will be reserved for servers that need to operate in a less secure network environment, such as the Ubuntu and Windows servers. The pfSense firewall will be operating within the WAN under the static IP Address: **10.161.21.4**.

A **DMZ (Demilitarized Zone)** is a local network that is separate from a business's private LAN. This isolated network enhances security by allowing external-facing services to communicate with the open internet while keeping the internal network protected.

VM Configuration

Guest OS: Free BSD 13 or later (64 bit)
Hardware: 2 CPUs, 8 GB RAM, Thin Provision, 3 Network Adapters Use Datastore ISO file
ISO File: **pfSense 2.7.2**

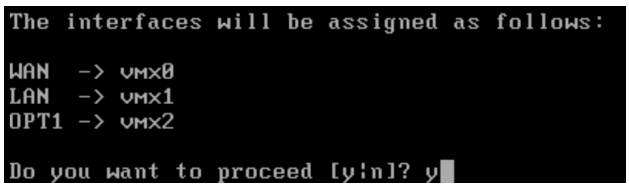
Final Specifications



pfSense Setup

Network Configuration

- Use **“zfs”** partitioning
    - Select the” VMware Virtual Disk”
- Enter NICs for the WAN, LAN, and DMZ
    - **WAN** = vmx0
    - **LAN** = vmx1
    - Optional 1 (**DMZ**) = vmx2



LAN (default to **192.168.1.1/24**)

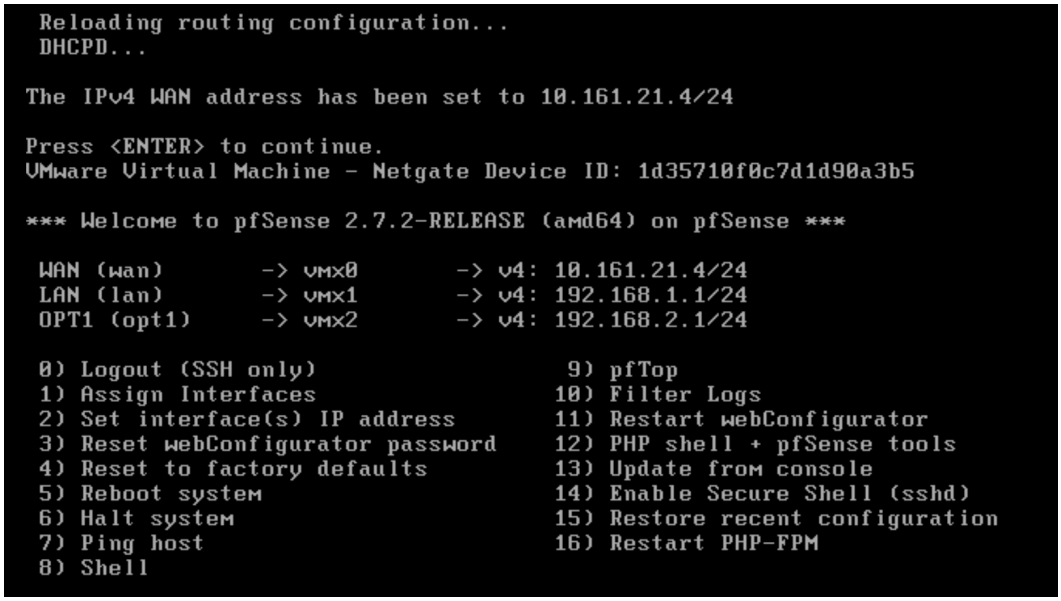
WAN Setup

- Adapter vmx0
- pfSense WAN Address set to: **10.161.21.4/24**
- WAN Upstream Gateway set to: **10.161.21.1**
- Set as **“Default Gateway”**
- No IPv6
- No DHCP on WAN

DMZ Setup

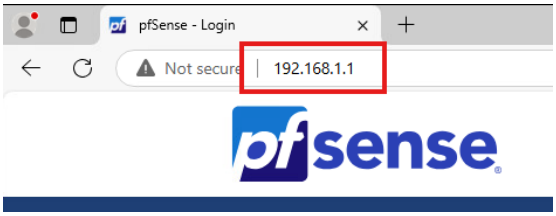
- Adapter vmx2
- DMZ Gateway address set to: **192.168.2.1/24**
- No WAN Usptream Address
- No IPv6
- No DHCP

Final Configuration:



pfSense Configuration

Navigate to 192.168.1.1 in a browser and sign in



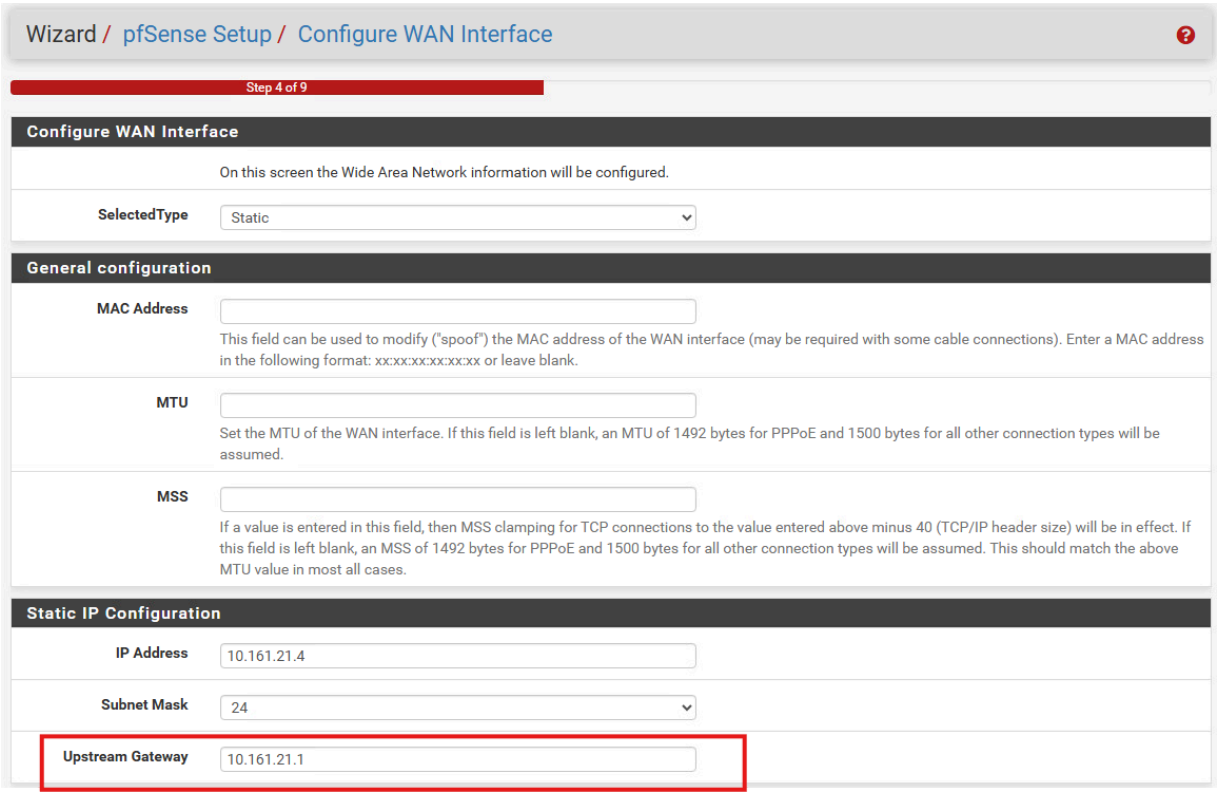
- Default Credentials
- Username: admin
- Password: pfsense

General Information

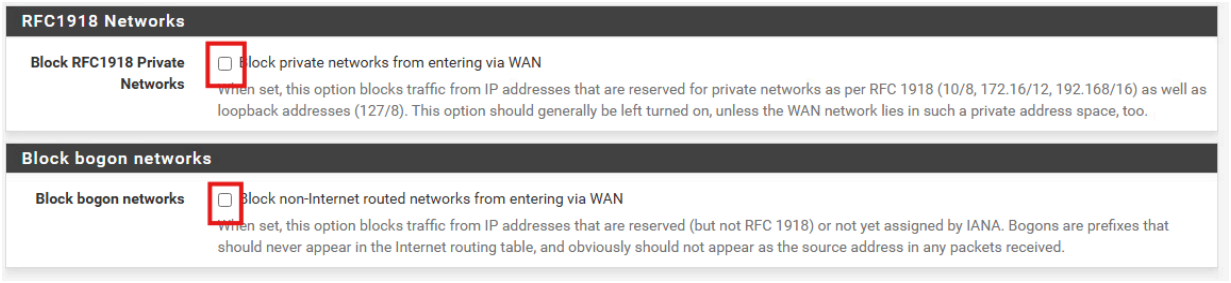
- Hostname: pfSense
- Domain: home.arpa
- Primary DNS Server: 10.120.16.10
- Secondary DNS Server: 10.120.16.11
- Override DNS: ☑️

Time Server (leave default)

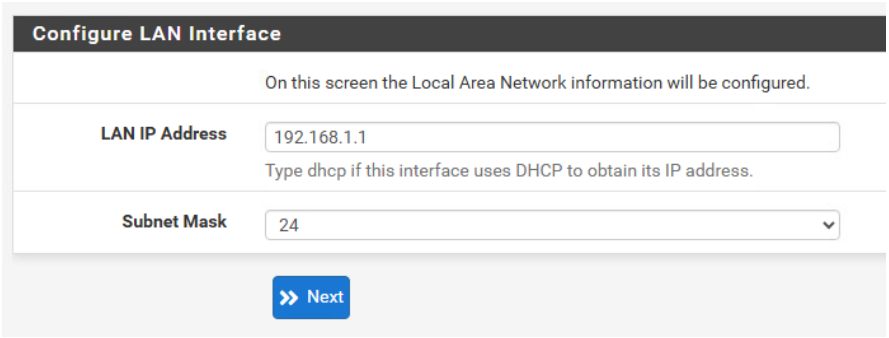
WAN Configuration



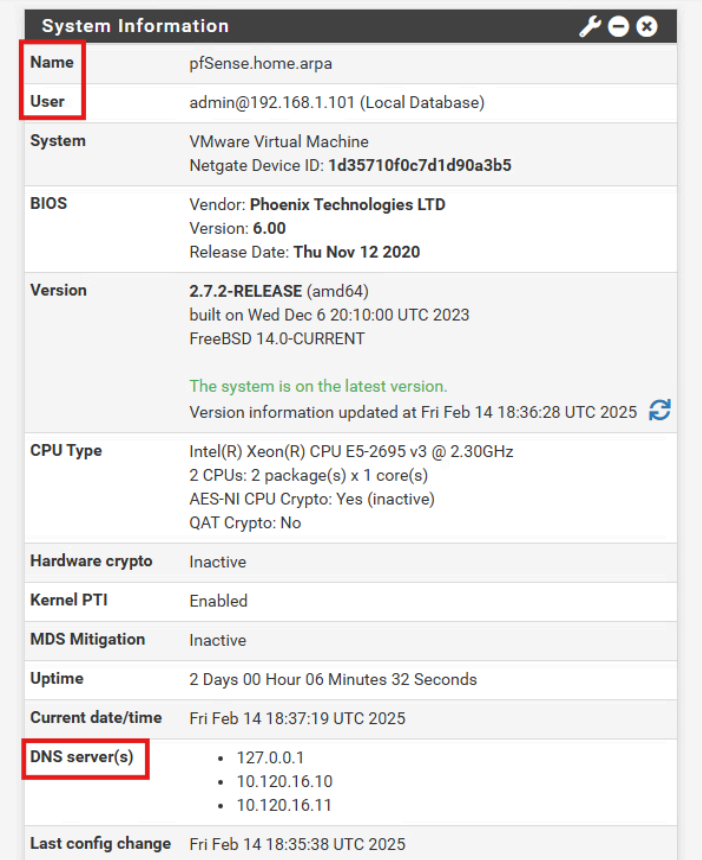
Uncheck **RFC1918 Networks** and **Bogon Networks**



LAN Configuration

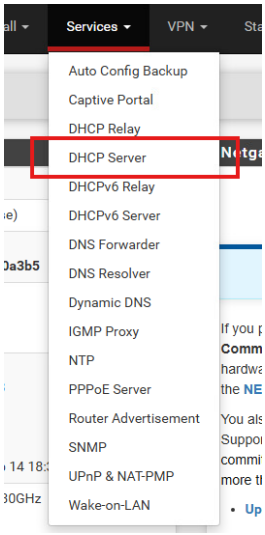


Final System Info:

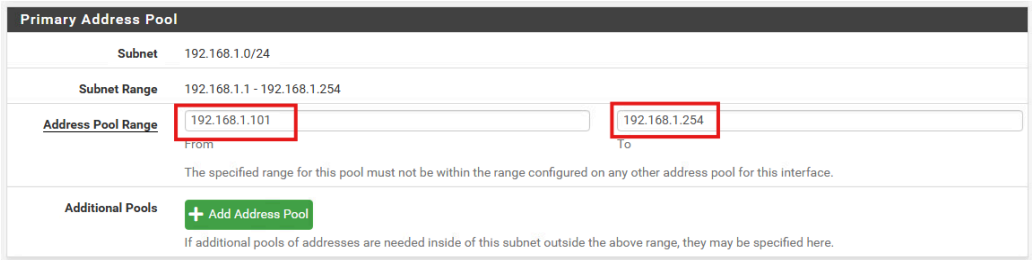


DHCP Setup

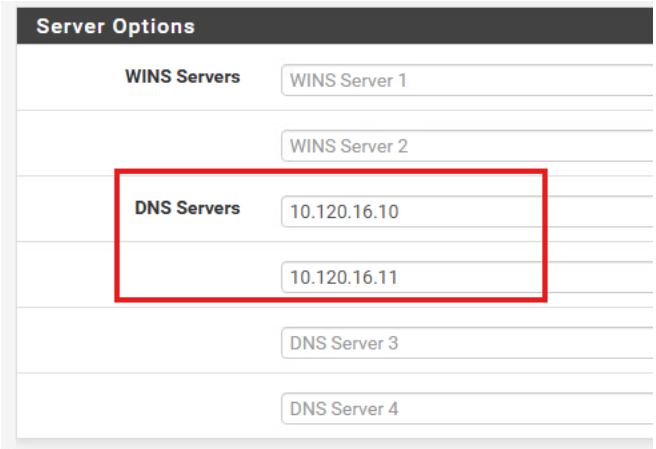
Edit the DHCP Server



Set DHCP Range: **192.168.1.101 to 192.168.1.254**

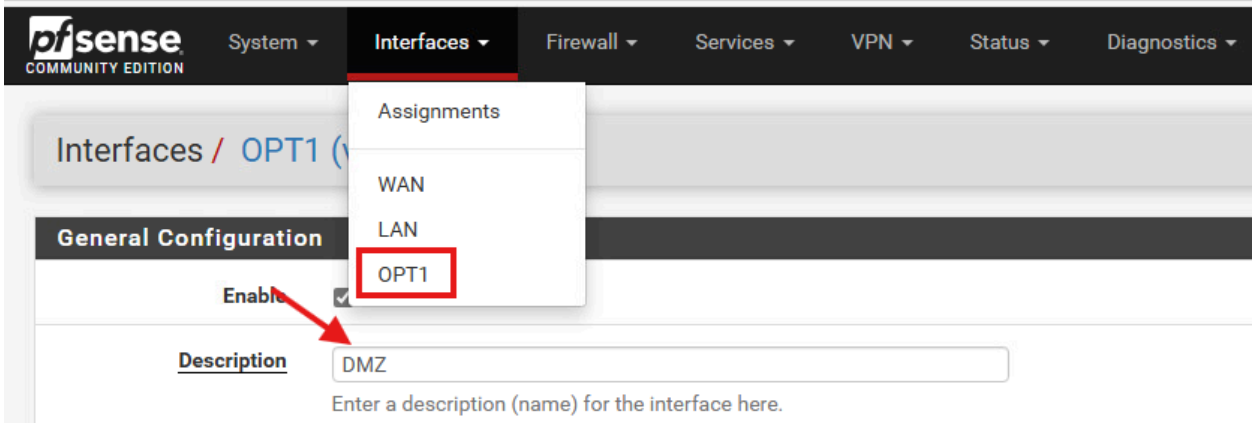


Configure LAN DNS



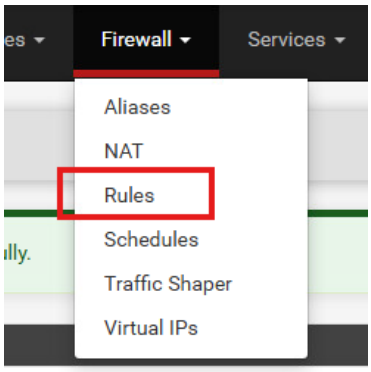
REMINDER: **Save** settings at the bottom of the screen and then **Apply Changes** at the top

Change to **OPT1** Interface and rename to **DMZ** -> **Save** and **Apply Changes**



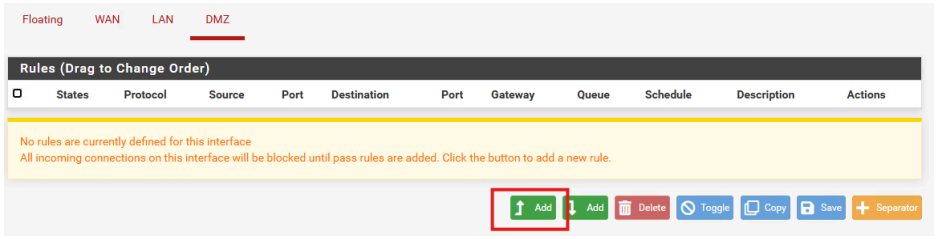
Firewall Rules

Navigate to Firewall menu

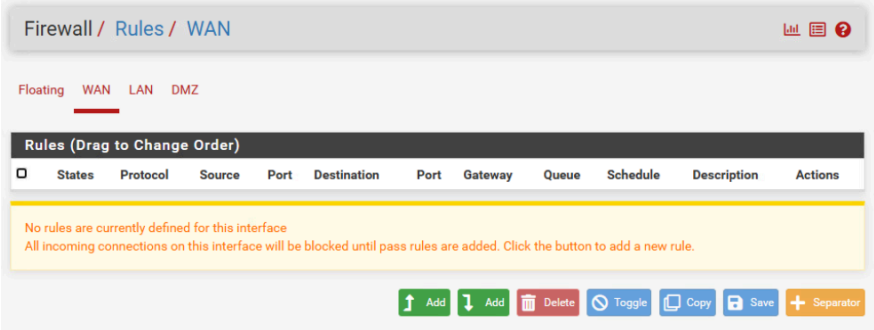


Use the bottom menu to **add**, **delete**, **edit**, and reorder firewall rules.

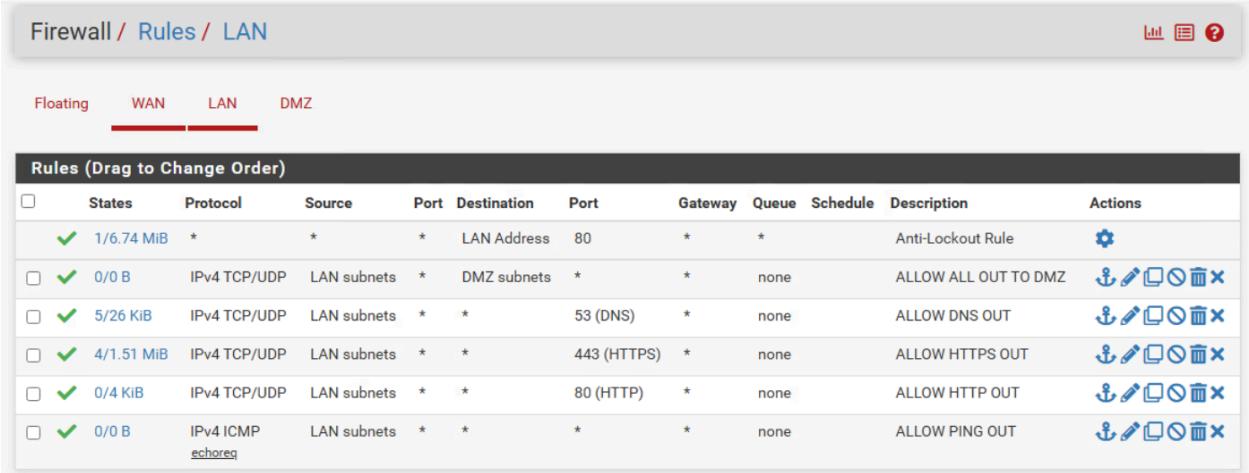
- NOTE: Order does matter, rules that are higher on the list have more priority



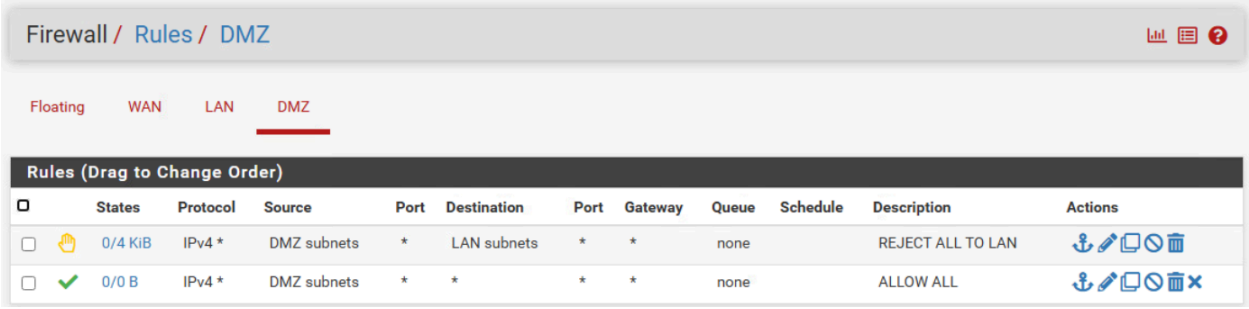
WAN Firewall Rules



LAN Firewall Rules



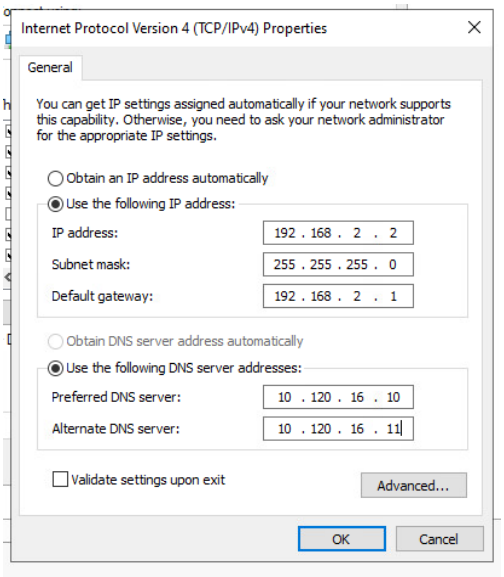
DMZ Firewall Rules



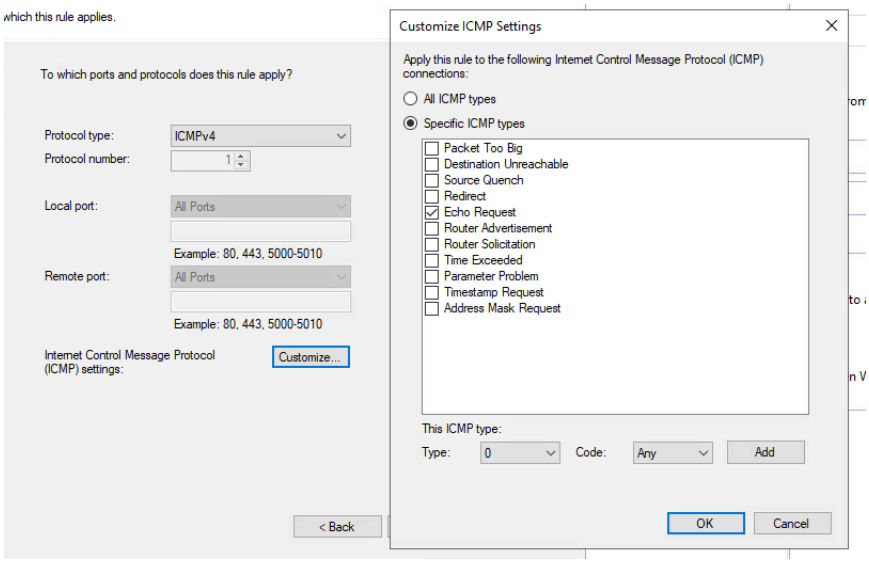
Internal Device Placement

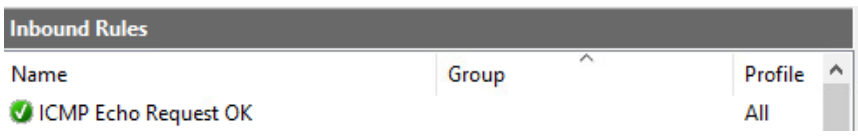
Windows Server (DMZ)

Go into Network and Internet Settings to edit the “Ethernet Adapter”
DMZ Static IP: **192.168.2.2**



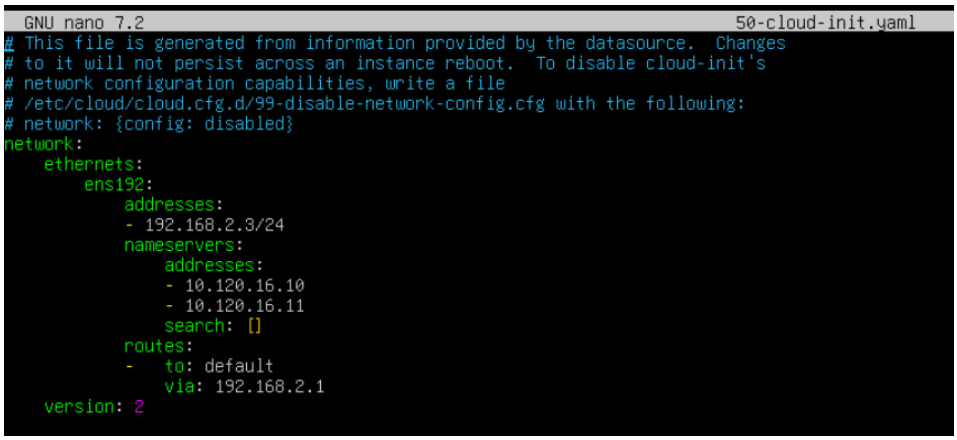
Add a firewall rule to allow **inbound pinging** of the Windows Server





Ubuntu Server (DMZ)

Change directories to /etc/netplan
Edit the network config file with: **sudo nano 50-cloud-init.yaml**
DMZ Static IP: **192.168.2.3**
Refresh the network: **sudo netplan apply**



Windows Client Setup (LAN)

Creating the following 2 Windows clients will simulate workstations in my company. They will be positioned in the LAN and the pfSense DHCP server should automatically issue out individual network configurations.

Windows Setup for Local Accounts (No Microsoft Email Required):

1. I disconnected the VM network card once each computer was in the final “windows setup” mode.
2. I then ran the following commands to bypass the required network setup so that I could create local system accounts:
    a. **Shift + f10** -> Brings up a command prompt
    b. Command: **OOBE\BYPASSNRO** (Out Of Box Experience command)
3. Once through setup without internet, reconnect the network adapter!

Windows 11 Workstation-01
Username: Bill
Password: **Classified**

Windows 11 Workstation-02
Username: Fernando
Password: **Classified**

NOTE: “Bob” is the answer for all security questions!!!

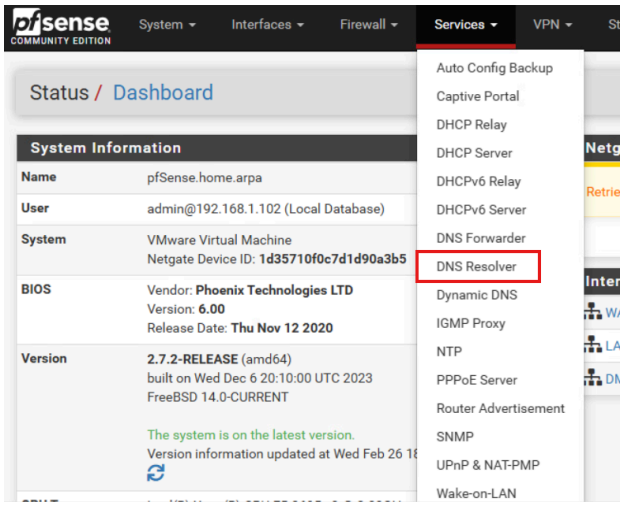
Project 4: Internal Caching DNS

Below contains steps outlining the configuration of an **internal caching DNS server** on my **pfSense firewall** which already manages DHCP. The internal DNS server will allow connected machines to have fast internet access due to IP caching. Additionally, its integration with the existing DHCP service will automatically configure DNS settings for LAN clients.

The **DNS Resolver** service within pfSense is easily integrated with the DHCP server. Having pfSense manage both allows clients connecting to the LAN to be issued an IP and the internal caching server simultaneously. The internal caching server also allows configuration of “static IPs” within the LAN for specific use cases. The IP address of the internal caching DNS Server is **192.168.1.1**. Since the DNS server is hosted on the firewall, DMZ clients can access it through the DMZ gateway address, which is **192.168.2.1**. This setup is beneficial as it allows the firewall to maintain a REJECT ALL traffic rule from the DMZ to the LAN.

DNS Resolver Configuration

To configure the internal DNS Resolver (Unbound) navigate to **Services > DNS Resolver**



Enable the DNS Resolver

Keep all ports default (Port 53)

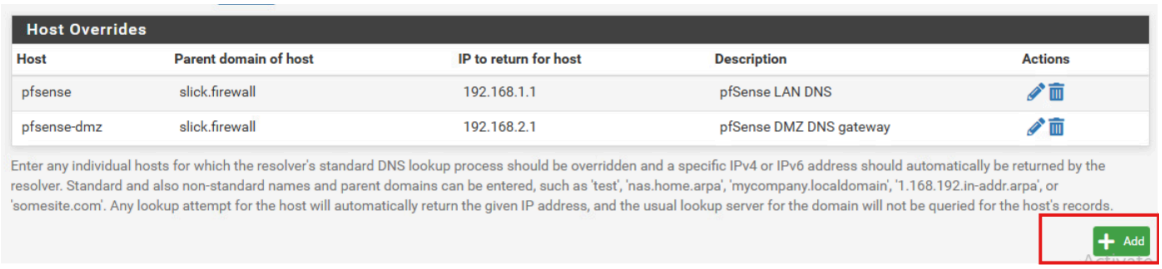
Set **Network Interfaces** and **Outgoing Network Interfaces** to “All”

Disable **DNSSEC** (does not integrate well with UNI DNS Servers)

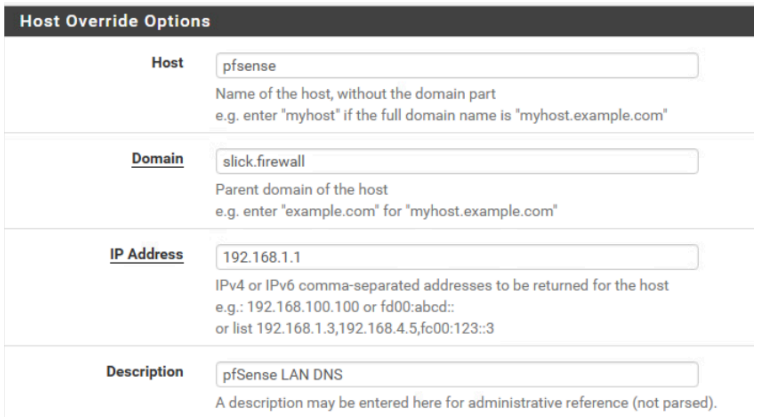
Enable **DNS Query Forwarding, DHCP Registration, Static DHCP**

Add “Host Overrides” at the bottom of the page for the LAN and DMZ

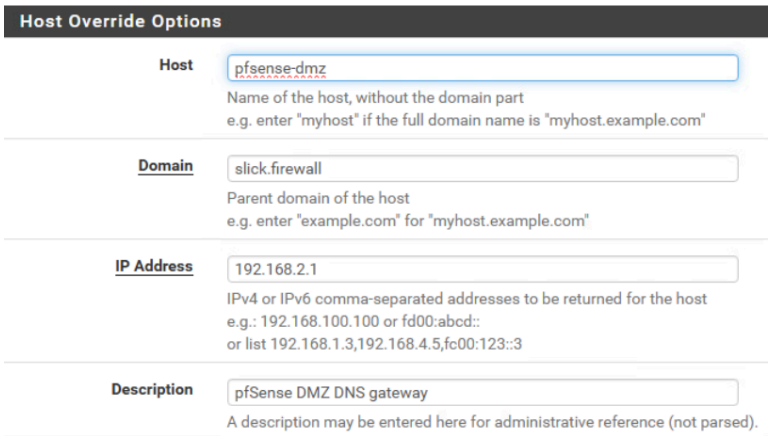
- This will allow the LAN and DMZ to fetch a hostname for the firewall for whichever gateway is being used by the client.



LAN: **pfsense.slick.firewall**

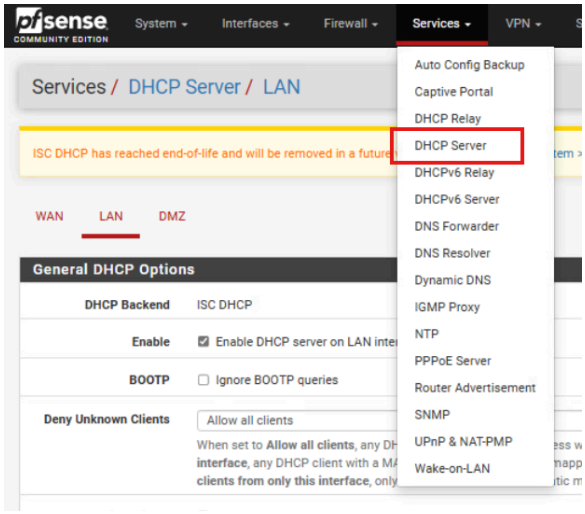


DMZ: **pfsense-dmz.slick.firewall**

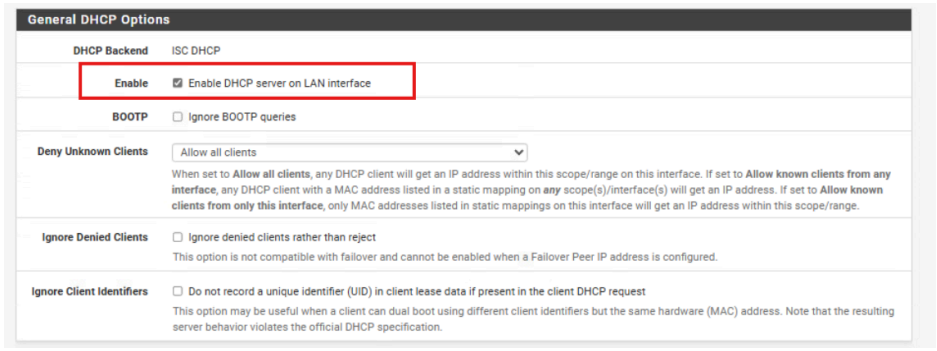


DHCP Configuration

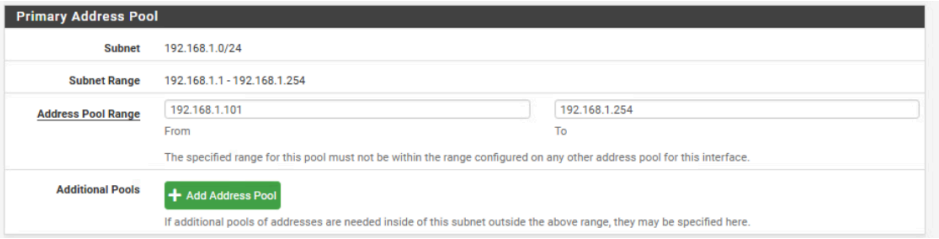
To configure DHCP navigate to **Services > DHCP Server**
Then make sure you are on the **LAN** tab.



Ensure DHCP is **enabled** on LAN



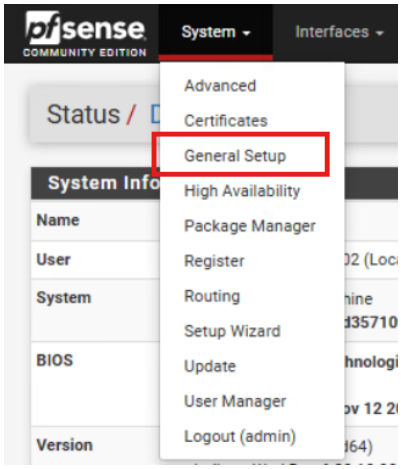
Ensure primary address pool is within the LAN subnet



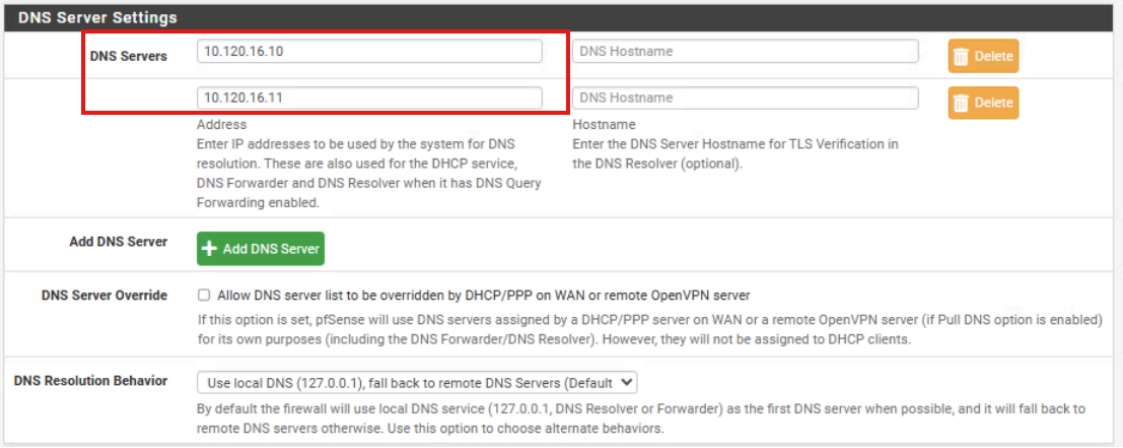
Configure the DHCP issued DNS Server to be the internal caching server: **192.168.1**

System Configuration

To ensure requests are forwarded from the internal caching server to the local recursive DNS servers navigate to **System > General Setup**



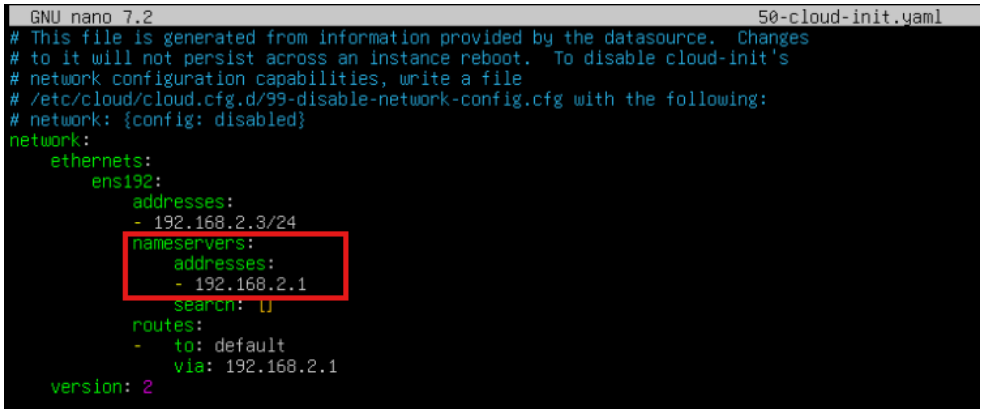
Ensure your Local Recursive DNS servers are properly configured
In this case I have the UNI DNS Servers: **10.120.16.10** and **10.120.16.11**



Ubuntu DNS Config

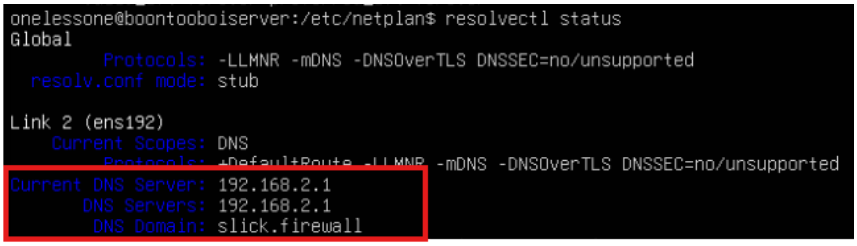
To connect the Ubuntu server statically set to 192.168.2.3 in the DMZ to the internal DNS hosted by the pfSense firewall do the following:

Edit the netplan file: **sudo nano /etc/netplan/50-cloud-init.yaml**
Replace the current name server addresses with **192.168.2.1**
Since the server is on the firewall, the DMZ gateway address should allow the server to utilize the internal DNS Resolver that was configured.

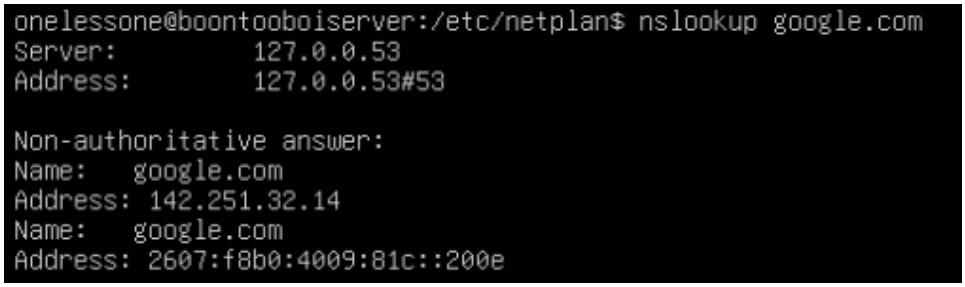


Refresh the network with the command: **sudo netplan apply**

Test your name server with the command: **resolvectl status**
It should output the internal caching DNS IP which is pfsense-dmz.slick.firewall:**192.168.1.1**



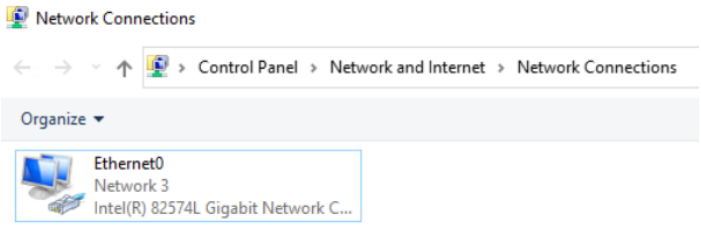
Test the DNS connection with the command: **nslookup google.com**



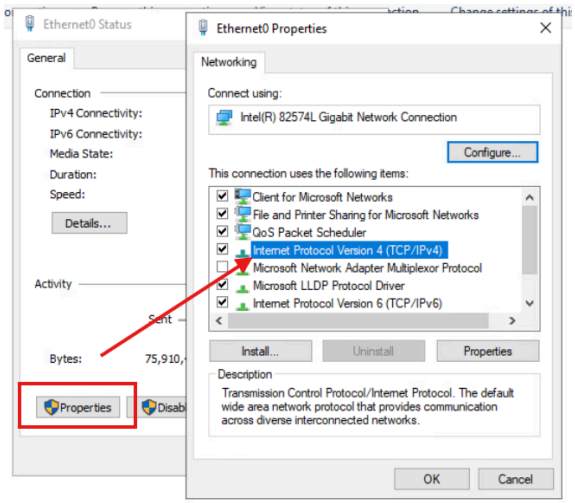
Windows DNS Config

To connect the Windows server statically set to 192.168.2.2 in the DMZ to the internal DNS hosted by the pfSense firewall do the following:

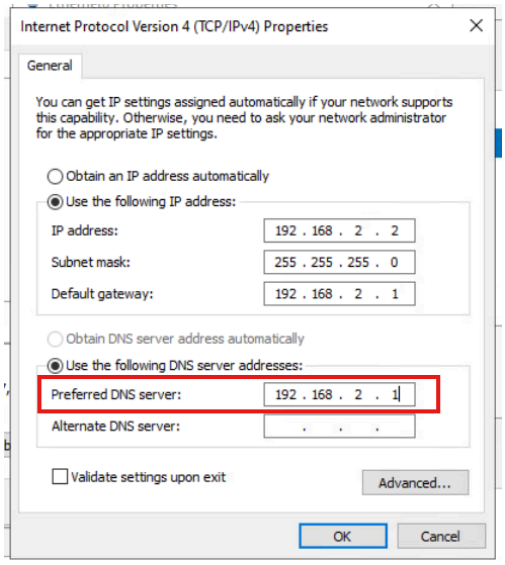
Navigate to edit the server's ethernet connection



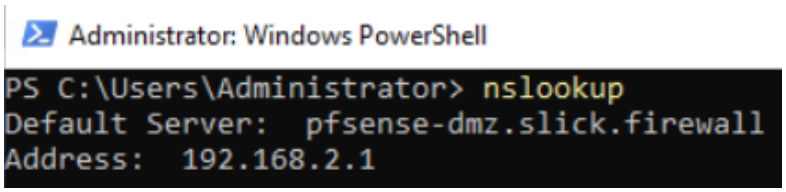
Edit the "Properties" of the adapter and then select "Internet Protocol Version 4"



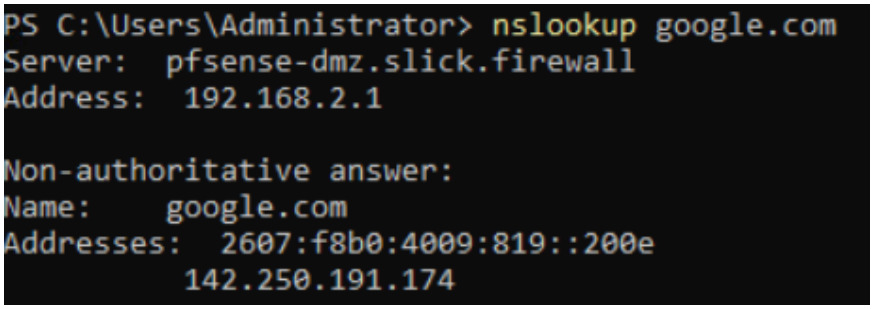
Set the "Preferred DNS server" to **192.168.2.1**



Test your name server with the command: **nslookup**
It should output the internal caching DNS IP which is pfsense-dmz.slick.firewall**:192.168.1.1**



Test the DNS connection with the command: **nslookup google.com**



LAN Client Testing

If you properly configured the internal caching DNS server, you should be able to do the following in a terminal from a LAN machine:

Test the functionality and verify the DNS Server is the correct address with the command: **nslookup google.com**

The DNS Address came back as **192.168.1.1** which is perfect because that is what was configured!

```
PS C:\Users\Bill> nslookup google.com
Server:  pfSense.slick.firewall
Address:  192.168.1.1

Non-authoritative answer:
Name:    google.com
Addresses:  2607:f8b0:4009:819::200e
          142.250.191.174
```

Test caching with the dig command: **dig mlb.com**
Run the command twice. The first query should be much longer than the second!

```
PS C:\Users\Bill> dig mlb.com

; <<>> DiG 9.16.28 <<>> mlb.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 51152
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1432
;; QUESTION SECTION:
;mlb.com.                       IN      A

;; ANSWER SECTION:
mlb.com.                3600    IN      A       34.102.163.158

;; Query time: 15 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Wed Mar 05 17:33:47 Pacific Standard Time 2025
;; MSG SIZE  rcvd: 52
```

```
PS C:\Users\Bill> dig mlb.com

; <<>> DiG 9.16.28 <<>> mlb.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 23439
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1432
;; QUESTION SECTION:
;mlb.com.                       IN      A

;; ANSWER SECTION:
mlb.com.                3589    IN      A       34.102.163.158

;; Query time: 0 msec
;; SERVER: 192.168.1.1#53(192.168.1.1)
;; WHEN: Wed Mar 05 17:33:58 Pacific Standard Time 2025
;; MSG SIZE  rcvd: 52
```

Project 5: External DNS

Below contains steps outlining the configuration of an **external DNS server**. The external DNS server will be hosted on the **Ubuntu Server** using **BIND**. This server will respond to queries on the public and private side of the firewall and will resolve requests made to the two servers in the DMZ (Windows and Ubuntu). Within the BIND configuration, FQDNs will be created and mapped to each DMZ server. Doing this will allow clients to query the DMZ servers by a domain name such as “www.mywebsite.com” instead of by IP address.

DNS Information
pfSense firewall: **10.161.21.4**

FQDN Mappings:
www.slickbrickcentral.com -> **192.168.2.3**
ubuntu.slickbrickcentral.com -> **192.168.2.3**
win.slickbrickcentral.com -> **192.168.2.2**

Ubuntu DNS Server

BIND (Berkeley Internet Name Domain) is a software that maps **FQDNs** (Fully Qualified Domain Names) to IP addresses. It is open source and can be configured to serve as an authoritative DNS server, a caching DNS server, or both. A **local authoritative DNS** server will be hosted under the static DMZ IP address: **192.168.2.3**.

BIND Installation

Run the following commands to install BIND:

1. **sudo apt update**
2. **sudo apt install bind9 bind9utils bind9-doc**

Configure BIND to use IPv4 protocol: **sudo nano /etc/default/named**
Add -4 to the end of the **OPTIONS** parameter:

Restart BIND to implement the changes: **sudo systemctl restart bind9**

BIND Configuration

Main Configuration File: **named.conf.options**
Open the config file: **sudo nano /etc/bind/named.conf.options**

If you want to forward unknown queries, uncomment and update the “forwarders” section with your DNS servers: **UNI DNS Servers**: **10.120.16.10** and **10.120.16.11**.

```
 GNU nano 7.2                                          /etc/bind/named.conf.options *
options {
        directory "/var/cache/bind";

        // If there is a firewall between you and nameservers you want
        // to talk to, you may need to fix the firewall to allow multiple
        // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

        // If your ISP provided one or more IP addresses for stable
        // nameservers, you probably want to use them as forwarders.
        // Uncomment the following block, and insert the addresses replacing
        // the all-0's placeholder.

        forwarders {
                10.120.16.10;
                10.120.16.11;
        };

        allow-query {
                any;
        };_

        //========================================================================
        // If BIND logs error messages about the root key being expired,
        // you will need to update your keys.  See https://www.isc.org/bind-keys
        //========================================================================
        dnssec-validation no;

        listen-on-v6 { any; };
};
```

I however do NOT want to forward queries. I want BIND to resolve only mappings I configure locally!

Add the following lines to <u>disable forwarding</u>

```
 GNU nano 7.2                                                  /etc/bind/named.conf.options
options {
        directory "/var/cache/bind";

        // If there is a firewall between you and nameservers you want
        // to talk to, you may need to fix the firewall to allow multiple
        // ports to talk.  See http://www.kb.cert.org/vuls/id/800113

        // If your ISP provided one or more IP addresses for stable
        // nameservers, you probably want to use them as forwarders.
        // Uncomment the following block, and insert the addresses replacing
        // the all-0's placeholder.

        //forwarders {
        //      10.120.16.10;
        //      10.120.16.11;
        //};

        forwarders {};

        allow-query {
                any;
        };

        //========================================================================
        // If BIND logs error messages about the root key being expired,
        // you will need to update your keys.  See https://www.isc.org/bind-keys
        //========================================================================
        dnssec-validation no;

        listen-on-v6 { any; };
};
```

Additional lines for logging in a new section below “options”

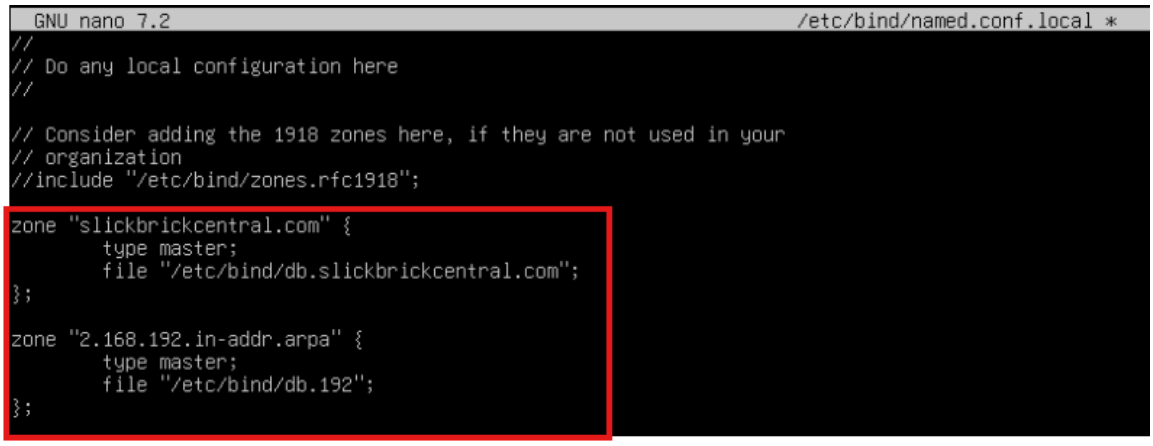
```
logging {
        channel query_log {
                file "/var/cache/bind/named.log" versions 3 size 5m;
                severity info;
                print-time yes;
                print-severity yes;
                print-category yes;
        };

        category queries { query_log; };
        category default { query_log; };
        category security { query_log; };

};
```

Access logs with the command: **sudo tail -f /var/cache/bind/named.log**

DNS Zones

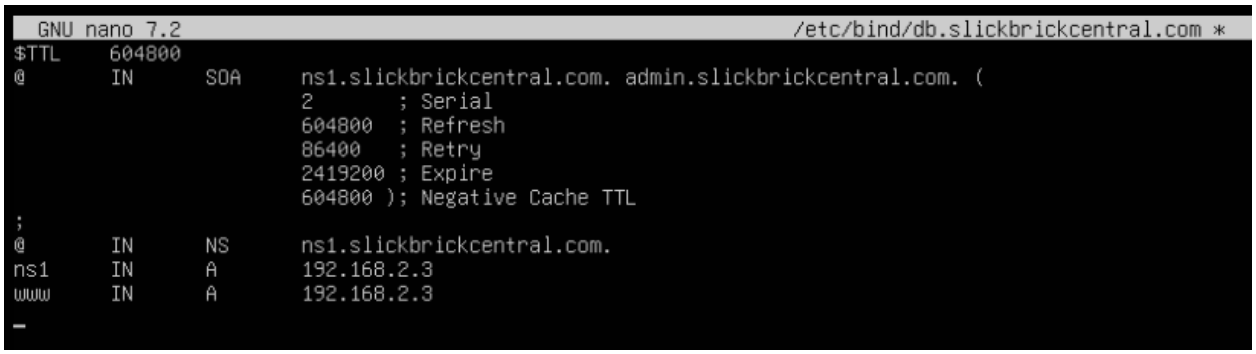
Edit the local configuration file to define your DNS Zones:
sudo nano /etc/bind/named.conf.local

This file specifies the DNS zones that BIND will manage.
I will be using ***slickbrickcentral.com*** as my company domain (you can change this out).
Also, since the DNS is in the DMZ subnet (192.168.2.0/24), I will use **2.168.192** in the second zone designation!



Forward Lookup Zone
Create the **forward lookup zone** file: **sudo nano /etc/bind/db.slickbrickcentral.com**

This file maps domain names to IP addresses.
Add the following content to the newly created file:



Reverse Lookup Zone

Create the reverse lookup zone file: **sudo nano /etc/bind/db.192**

This file maps IP addresses back to domain names.

Add the following content to the newly created file:

```
  GNU nano 7.2                                                    /etc/bind/db.192 *
$TTL    604800
@       IN      SOA     ns1.slickbrickcentral.com. admin.slickbrickcentral.com. (
                        2       ; Serial
                        604800  ; Refresh
                        86400   ; Retry
                        2419200 ; Expire
                        604800 ); Negative Cache TTL
;
@       IN      NS      ns1.slickbrickcentral.com.
3       IN      PTR     www.slickbrickcentral.com.
```

Check for Errors

Check the config files for syntax errors:

sudo named-checkconf

sudo named-checkzone slickbrickcentral.com /etc/bind/db.slickbrickcentral.com

sudo named-checkzone 2.168.192.in-addr.arpa /etc/bind/db.192

If there are no errors the command output should be as follows:

```
onelessone@boontooboiserver:/etc/netplan$ sudo named-checkconf
onelessone@boontooboiserver:/etc/netplan$ sudo named-checkzone slickbrickcentral.com /etc/bind/db.slickbrickcentral.com
zone slickbrickcentral.com/IN: loaded serial 2
OK
onelessone@boontooboiserver:/etc/netplan$ sudo named-checkzone 2.168.192.in-addr.arpa /etc/bind/db.192
zone 2.168.192.in-addr.arpa/IN: loaded serial 2
OK
```

Restart BIND9 to apply changes: **sudo systemctl restart bind9**

Configure UFW to allow DNS traffic (if applicable): **sudo ufw allow Bind9**

Test DNS with a dig command: **dig @localhost www.slickbrickcentral.com**

```
onelessone@boontooboiserver:/etc/netplan$ dig @localhost www.slickbrickcentral.com

; <<>> DiG 9.18.30-0ubuntu0.24.04.2-Ubuntu <<>> @localhost www.slickbrickcentral.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20642
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: cb8715302e6b6a940100000067ccad4b3be4bffaf3940bf5 (good)
;; QUESTION SECTION:
;www.slickbrickcentral.com.     IN      A

;; ANSWER SECTION:
www.slickbrickcentral.com. 604800 IN    A       192.168.2.3

;; Query time: 1 msec
;; SERVER: 127.0.0.1#53(localhost) (UDP)
;; WHEN: Sat Mar 08 20:49:15 UTC 2025
;; MSG SIZE  rcvd: 98
```

FQDN Records

To add **FQDN** records for the two static DMZ machines, we will have to update the forward and reverse zone files!

Update Forward Zone

Edit the forward lookup zone file: **sudo nano /etc/bind/db.slickbrickcentral.com**

Add **A record** for the machines:

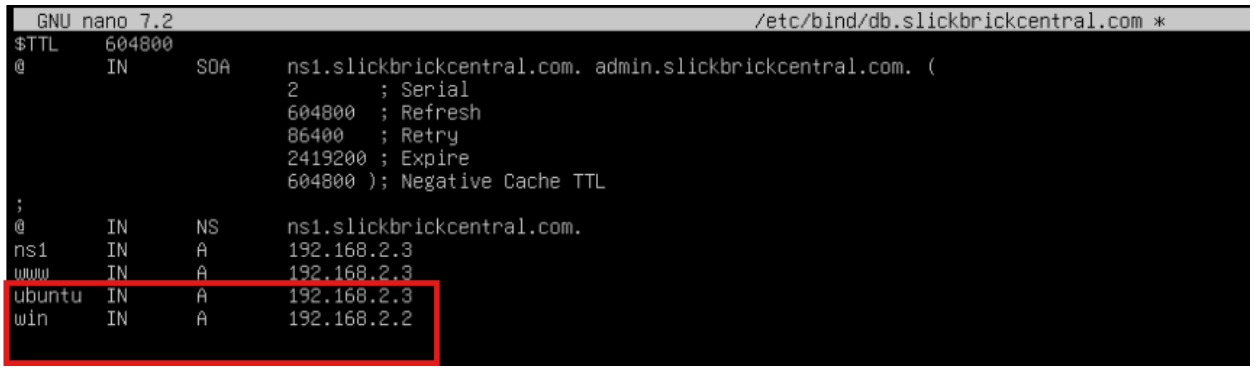
Ubuntu Server

- FQDN = ubuntu.slickbrickcentral.com
- IP: 192.168.2.3

Windows Server 2022

- FQDN = win.slickbrickcentral.com
- IP: 192.168.2.2

The file should look like this:



Update Reverse Zone

Edit the reverse lookup zone file: **sudo nano /etc/bind/db.192**

Add PTR (Pointer) records for the machines:

Ubuntu Server

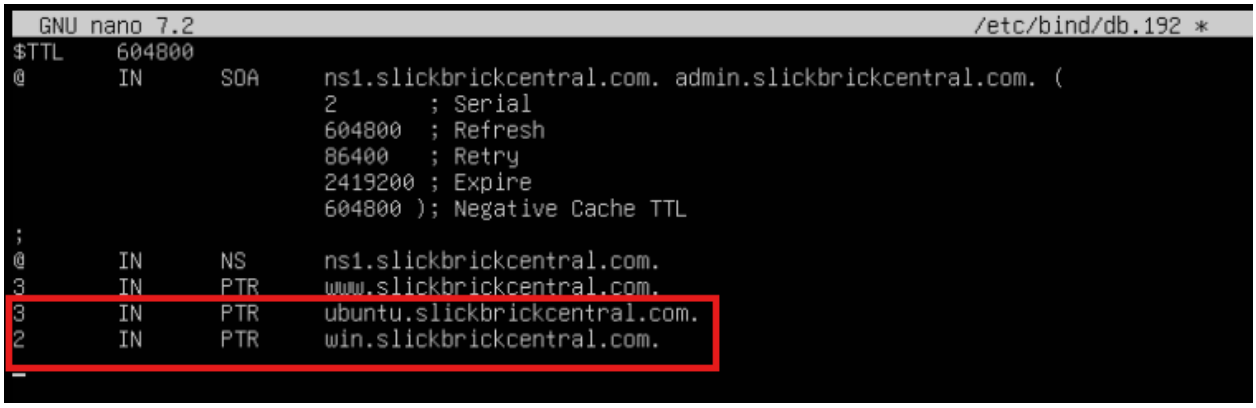
- FQDN = ubuntu.slickbrickcentral.com
- IP: 192.168.2.**3**

Windows Server 2022

- FQDN = win.slickbrickcentral.com
- IP: 192.168.2.**2**

NOTE: Add the last octet of the IP to the first entry in each PTR record!
For example: 192.168.2.55 → **55** IN PTR yay.slickbrickcentral.com

The file should look like this:



Check the config files for syntax errors:
sudo named-checkconf
sudo named-checkzone slickbrickcentral.com /etc/bind/db.slickbrickcentral.com
sudo named-checkzone 2.168.192.in-addr.arpa /etc/bind/db.192

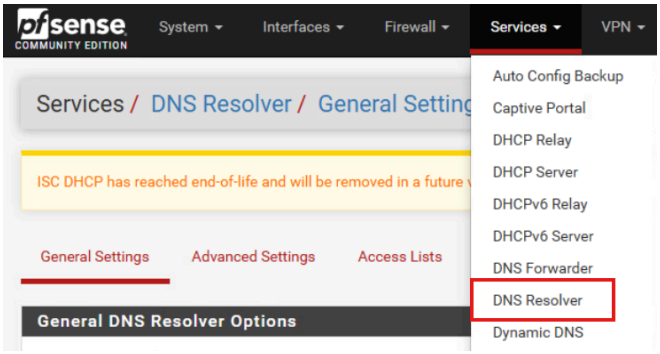
Restart BIND9 to apply changes: **sudo systemctl restart bind9**

pfSense Integration

In order to have your caching DNS server hosted on the pfSense firewall forward to the local DNS you just created, you need to configure pfSense to forward requests to 192.168.2.3 for your custom domain: slickbrickcentral.com.

Do this by going to the pfSense browser portal: https://192.168.1.1

Navigate to **Services > DNS Resolver** in the top bar and scroll to the bottom of the page.



Next, locate "Domain Overrides" at the bottom of the page.

You will need to add 2 records so your local FQDNs will resolve. One record will be the forward lookup zone and the other will be the reverse lookup zone.

Referencing the earlier DNS configuration:

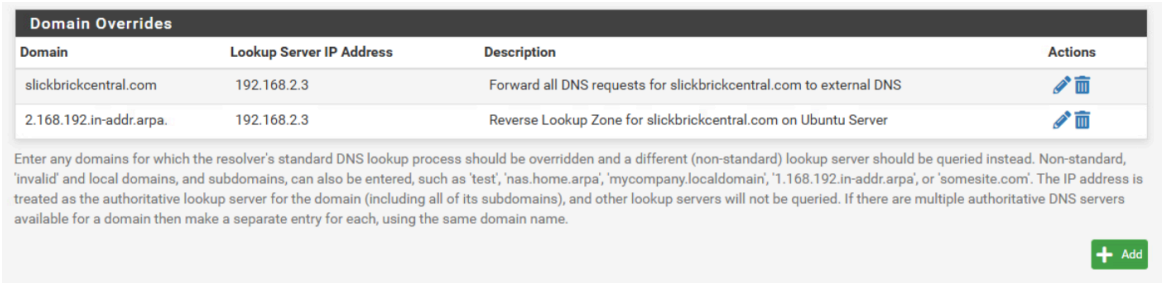
Forward Zone:

- slickbrickcentral.com
- 192.168.2.3

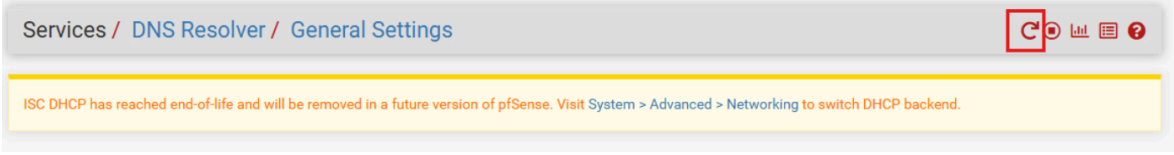
Reverse Zone:

- 2.168.192.in-addr.apra.
- 192.168.2.3

Add both rules:



Note: Make sure to select **Apply Changes** on the top of the page to lock in your changes. Then refresh the DNS Resolver service →



Test the Domain Overrides by using a LAN Client to **ping**, **nslookup**, and **dig** one of the FQDNs you created.

I will test win.slickbrickcentral.com -> 192.168.2.2

Run the command: **ping win.slickbrickcentral.com**

```
PS C:\Users\Bill> ping win.slickbrickcentral.com

Pinging win.slickbrickcentral.com [192.168.2.2] with 32 bytes of data:
Reply from 192.168.2.2: bytes=32 time<1ms TTL=127
Reply from 192.168.2.2: bytes=32 time<1ms TTL=127
Reply from 192.168.2.2: bytes=32 time<1ms TTL=127
Reply from 192.168.2.2: bytes=32 time<1ms TTL=127

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Run the command: **nslookup win.slickbrickcentral.com**

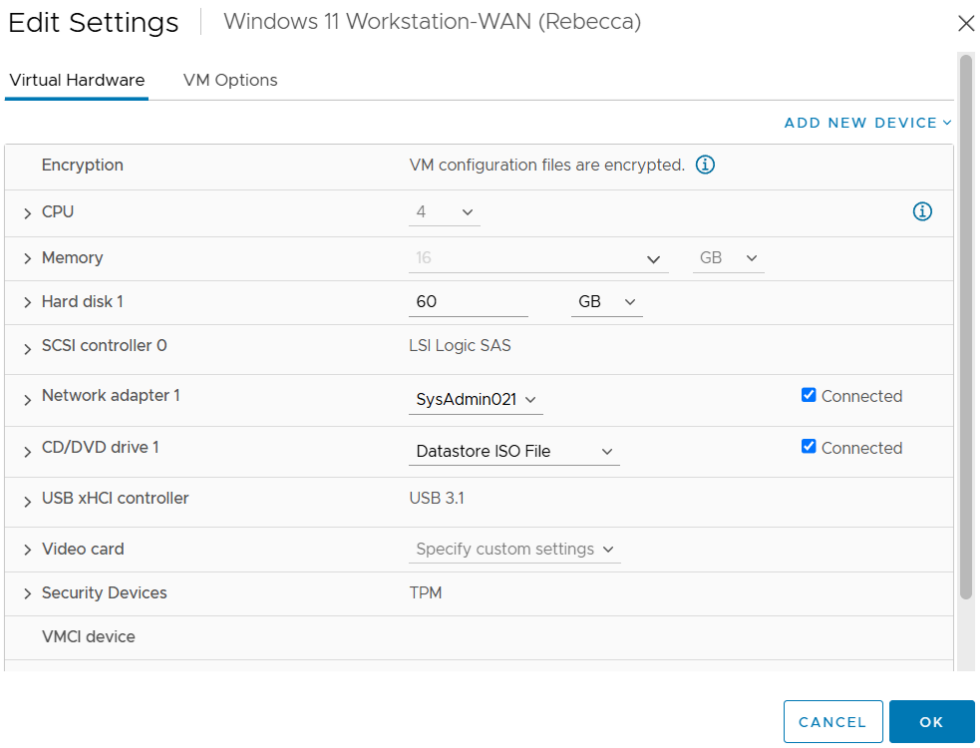
```
PS C:\Users\Bill> nslookup win.slickbrickcentral.com
Server:  pfSense.slick.firewall
Address:  192.168.1.1

Non-authoritative answer:
Name:    win.slickbrickcentral.com
Address:  192.168.2.2
```

Run the command: **dig win.slickbrickcentral.com**

WAN Configuration

VM Configuration

REMEMBER: **Thin Provision Storage** and **TPM Module**!!!



Windows Setup for Local Accounts (No Microsoft Email Required):

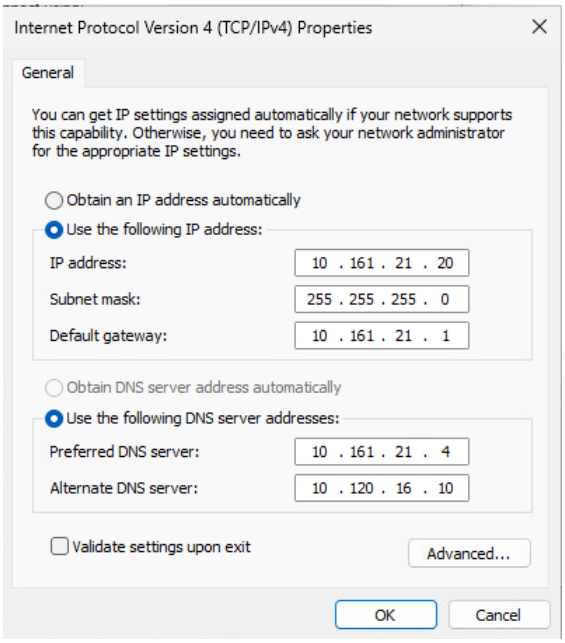
1. I disconnected the VM network card once each computer was in the final “windows setup” mode.
2. I then ran the following commands to bypass the required network setup so that I could create local system accounts:
    a. **Shift + f10** -> Brings up a command prompt
    b. Command: **OOBE\BYPASSNRO** (Out Of Box Experience command)
3. Once through setup without internet, reconnect the network adapter!

Windows 11 Workstation-WAN
Username: Rebecca
Password: **Classified**

NOTE: “Bob” is the answer for all security questions!!!

Workstation Network Configuration

Workstation IP: **10.161.21.20**
DNS Servers: **10.161.21.4** and **10.120.16.10**



I also installed the **Chocolatey** package manager and the dig command.
Powershell: **choco install bind-toolsonly**

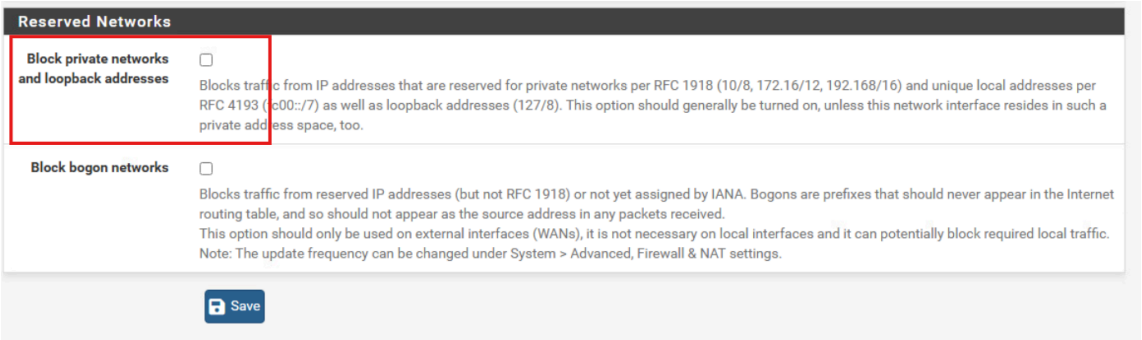
pfSense Address (WAN): **10.161.21.4**

I am going to forward any request from WAN to pfSense on port 53 straight to my “external DNS server”: **192.168.2.3** which then will reply back to the WAN client.

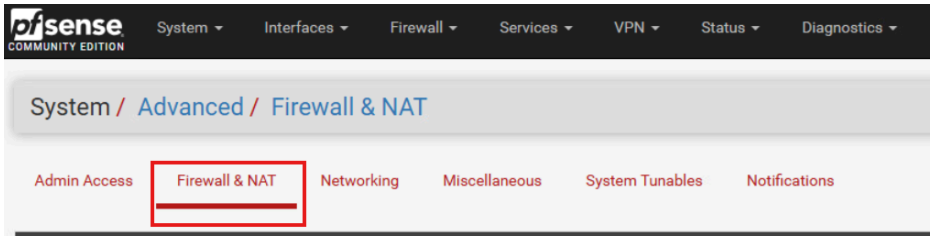
pfSense NAT

In the admin web interface: http://192.168.1.1

Navigate to **Interfaces > WAN**. Scroll to the bottom of the page and make sure that "Block private networks" is unchecked!

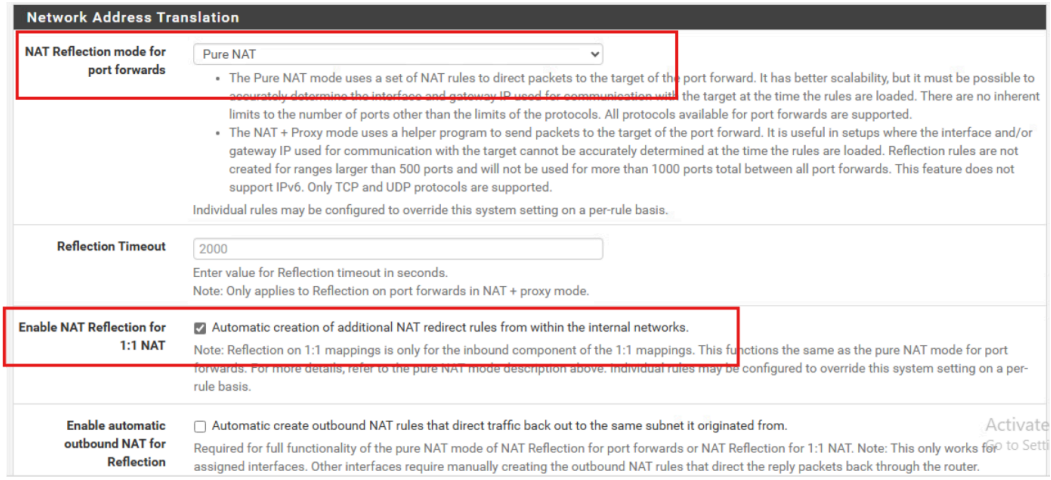


Navigate to **System > Advanced > Firewall and NAT** Tab.



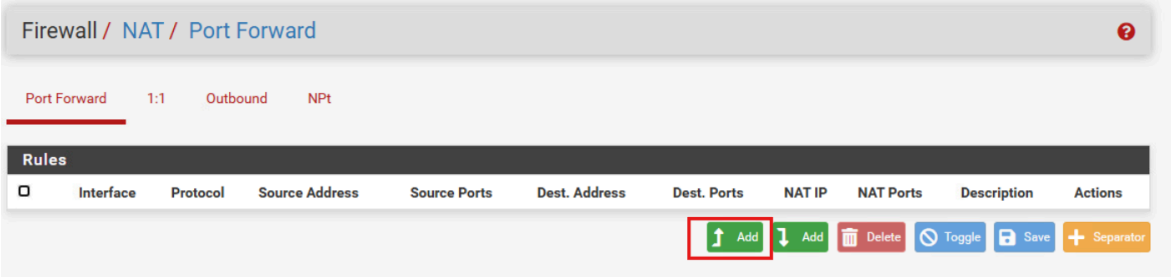
In the "Network Address Translation" section further down the page:

- Set "NAT Reflection Mode for Port Forwards" to **Pure NAT**
- **Enable** (Check the box) for "Enable NAT Reflection for 1:1 NAT"



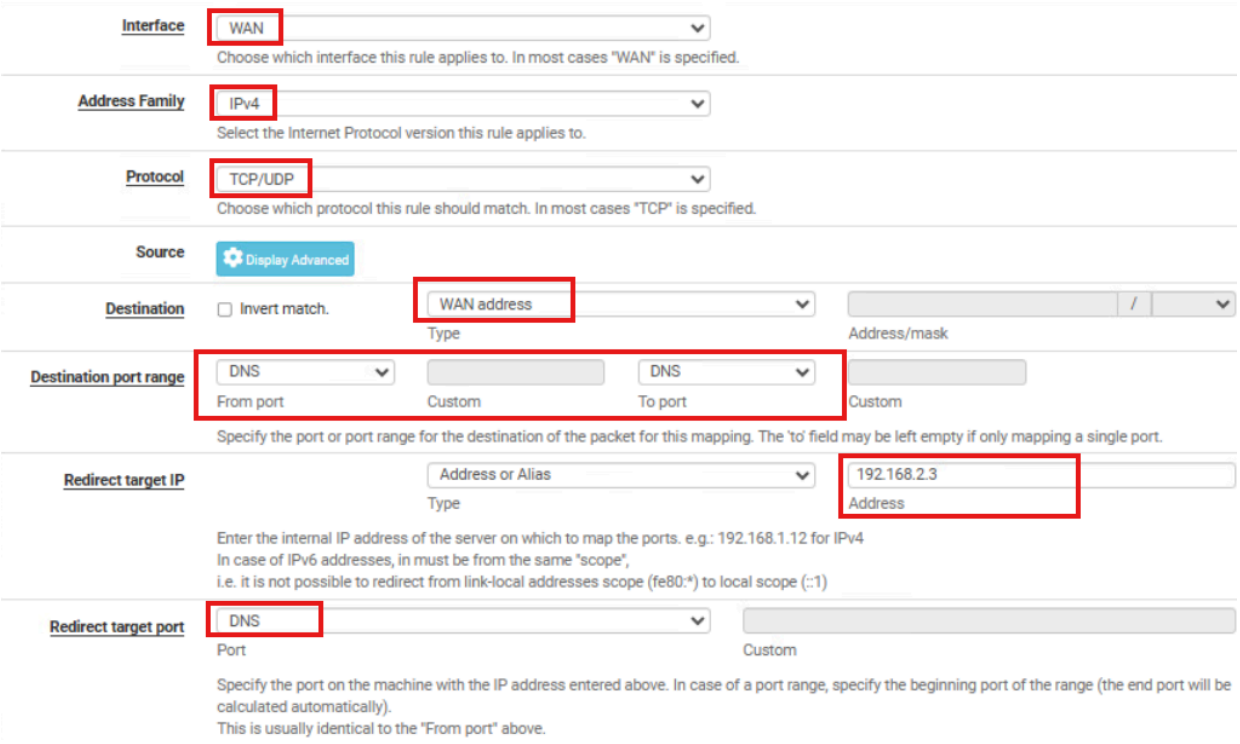
NAT Port Forwarding Rule

Navigate to **Firewall > NAT > Port Forward** Tab.

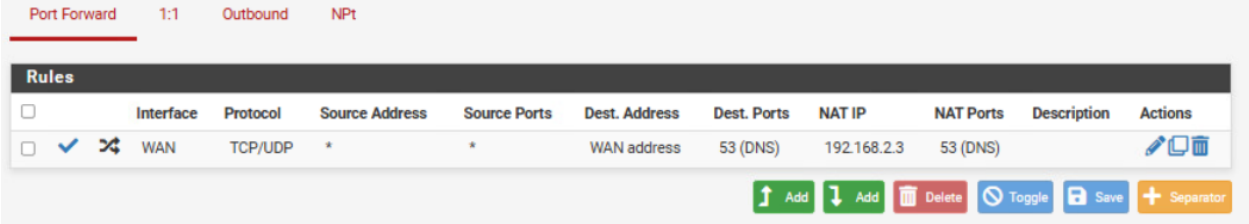


Add a new Rule with following configurations:

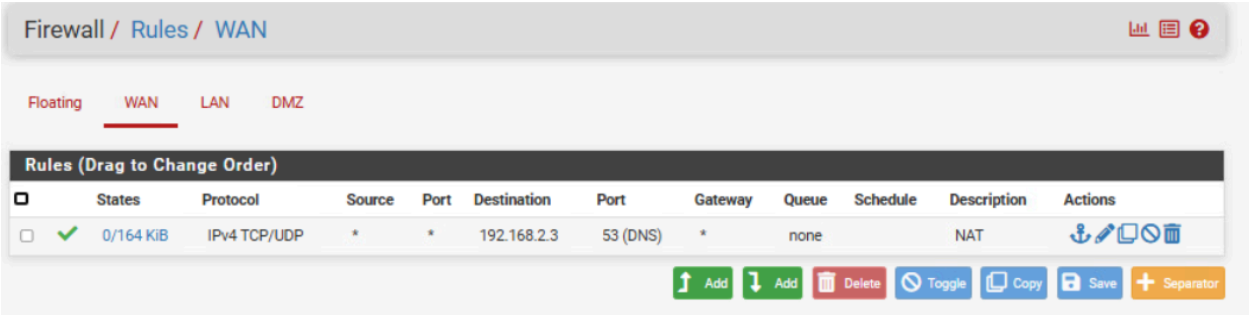
- **Interface:** WAN
- **Protocol:** TCP/UDP
- **Destination:** WAN Address
- **Destination Port Range:** DNS (Port 53 Only)
- **Redirect Target IP:** Internal DNS Server (BIND) → 192.168.2.3
- **Redirect Target Port:** DNS (Port 53)



Final NAT Rule:



Corresponding Firewall Rule Created:

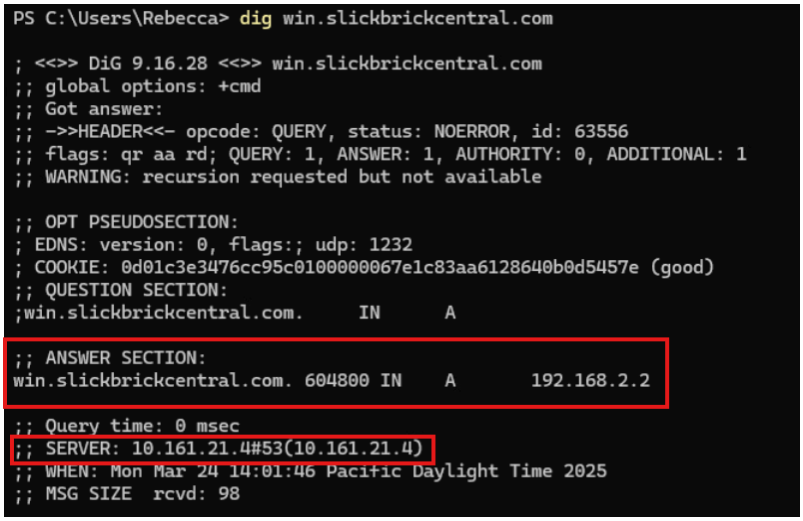


WAN Client Testing

Go to the WAN Client (10.161.21.20) created earlier and use the dig command to verify proper configuration and local DNS resolution.

First, we will show that DNS requests are properly being port forwarded from pfSense (**10.161.21.4**) to FQDNs mapped in the local BIND server (**192.168.2.3**).

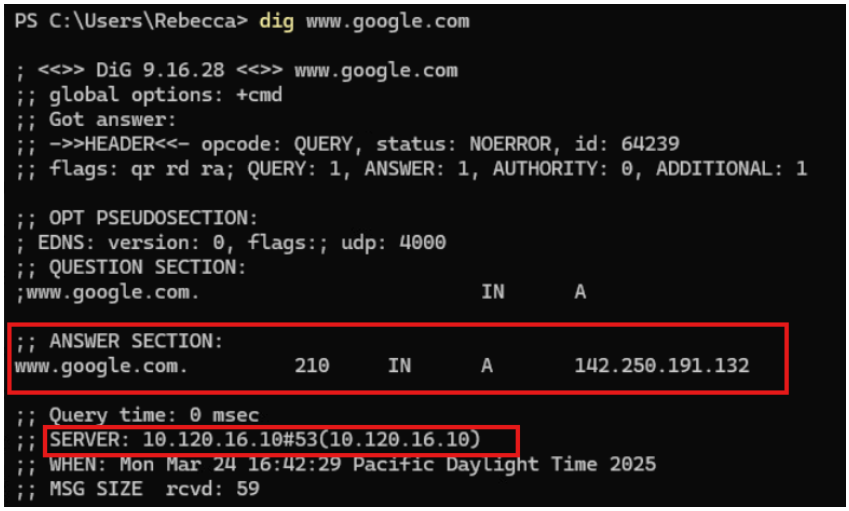
Run the command: **dig win.slickbrickcentral.com**



As you can see above, the **A record** for win.slickbrickcentral.com is 192.168.2.2 which is correct!

Next, we will show that requests **NOT** mapped in the BIND server are redirected to the alternate UNI DNS (**10.120.16.10**) configured on the WAN Machine.

Run the command: **dig google.com**



Project 6: Static HTTPS Web Server

Below contains steps outlining the configuration of a static **HTTPS** server using **Apache** on the existing **Ubuntu Linux** server (**192.168.2.3**) within the local DMZ. This server will server static HTML content to the frontend so no scripting language or database will be needed. Proper configuration of DNS records, a self signed certificate for SSL connections, and integration of **NAT port forwarding** in pfSense will allow clients in the LAN and WAN to access the web server.

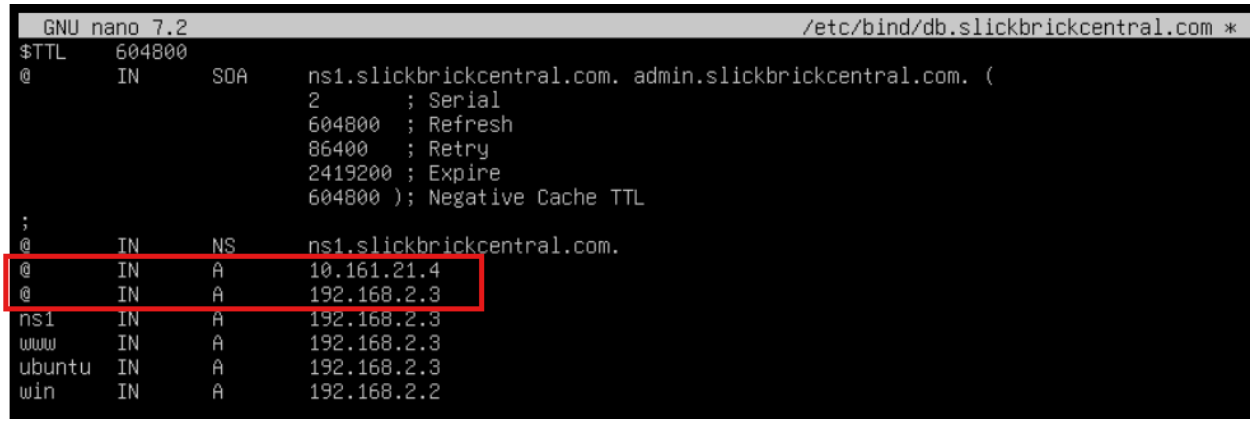
Web Server mapping:
slickbrickcentral.com and **www.slickbrickcentral.com** → **192.168.2.3** (Ubuntu Linux Server)

BIND DNS Records

Edit the forward lookup zone to add A records for slickbrickcentral.com.

Edit the configuration file: **sudo nano /etc/bind/db.slickbrickcentral.com**

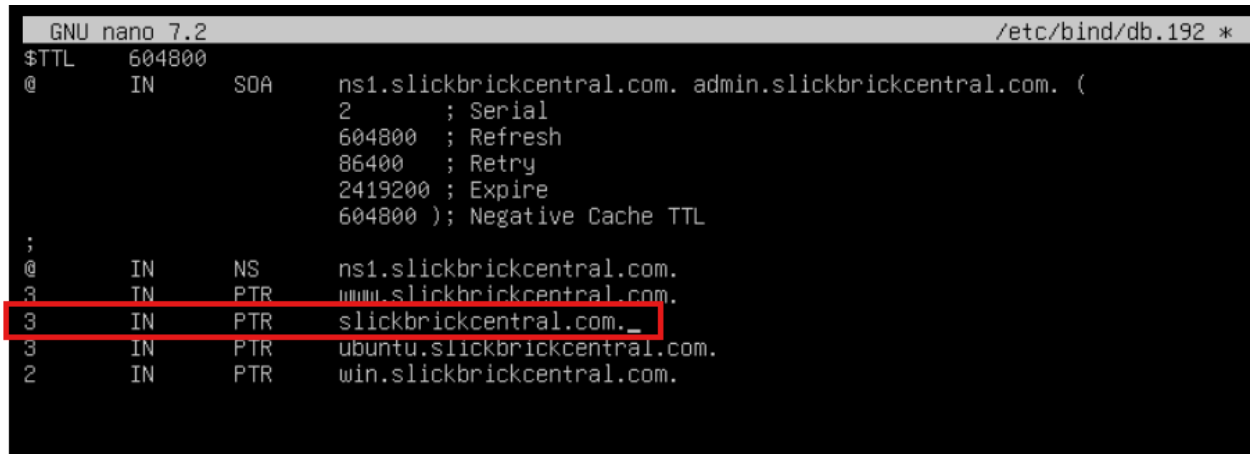
Add the following A Records:



Next update the reverse lookup zone.

Edit the configuration file: **sudo nano /etc/bind/db.192**

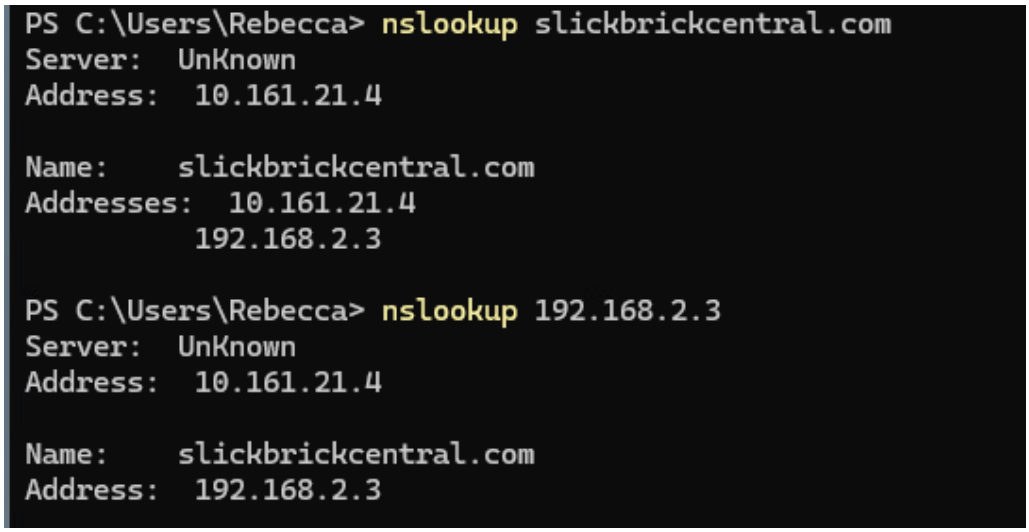
Add the following reverse lookup records:



Apply changes with the command: **sudo systemctl restart bind9**

Verify Records on WAN Client

Run the command: **nslookup slickbrickcentral.com**
Run the command: **nslookup 192.168.2.3**



Apache Web Server

The **Apache HTTP Server**, commonly known as Apache, is a powerful, open-source web server software that allows websites to be hosted and served over the internet. It is highly customizable and supports a wide range of features, including SSL/TLS encryption, URL redirection, and load balancing.

Install and Configuration

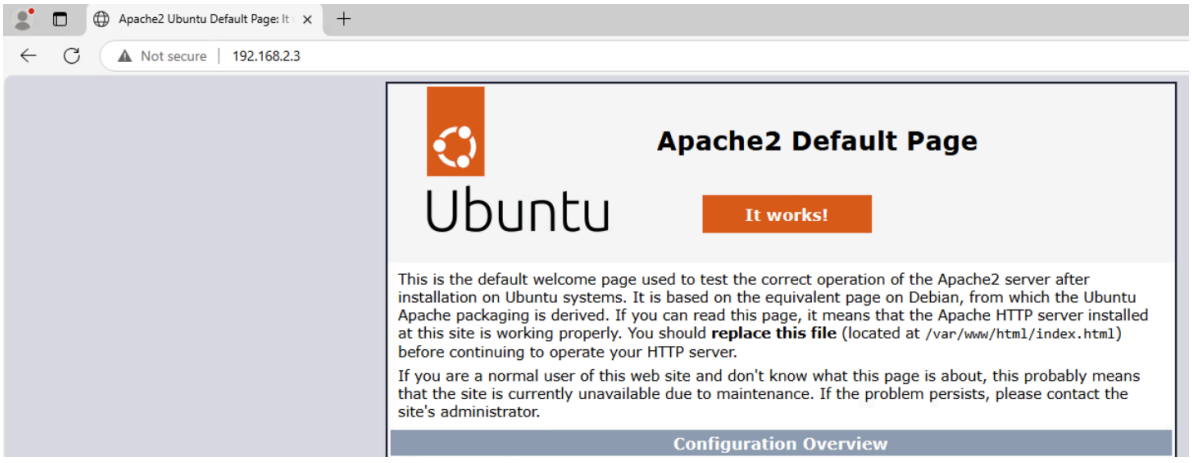
I will be installing Apache on the Ubuntu Linux Server in the DMZ under the IP 192.168.2.3.

Update system packages with: **sudo apt update**

To begin the installation run the command: **sudo apt install apache2**
Start the service with the command: **sudo systemctl status apache2**
Verify that it is running with the command: **sudo systemctl status apache2**

```
onelessone@boontooboiserver:~$ sudo systemctl start apache2
onelessone@boontooboiserver:~$ sudp systemctl status apache2
Command 'sudp' not found, did you mean:
  command 'ssdp' from snap ssdp (0.0.1)
  command 'sfdp' from deb graphviz (2.42.2-9ubuntu0.1)
  command 'sudo' from deb sudo (1.9.14p2-1ubuntu1)
  command 'sudo' from deb sudo-ldap (1.9.14p2-1ubuntu1)
  command 'sup' from deb sup (20100519-3)
See 'snap info <snapname>' for additional versions.
onelessone@boontooboiserver:~$ sudo systemctl start apache2
onelessone@boontooboiserver:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-03-28 17:30:53 UTC; 3min 19s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 144898 (apache2)
      Tasks: 55 (limit: 9445)
     Memory: 5.4M (peak: 5.9M)
        CPU: 72ms
     CGroup: /system.slice/apache2.service
             ├─144898 /usr/sbin/apache2 -k start
             ├─144901 /usr/sbin/apache2 -k start
             └─144902 /usr/sbin/apache2 -k start
```

Since the Ubuntu Server is within the DMZ, verify the website is being hosted locally by using a DMZ client and navigating to http://192.168.2.3 in a browser.



SSL Configuration

In order to ensure that our website connection is secure, we need to enable **SSL** so our network traffic is HTTPS. To enable SSL we can create a **self-signed certificate**.

Enable SSL Module

We will start by enabling an Apache module that provides support for SSL encryption.
Use the command: **sudo a2enmod ssl**
Restart Apache to activate the module: **sudo systemctl restart apache2**

```
onelessone@boontooboiserver:~$ sudo a2enmod ssl
[sudo] password for onelessone:
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
onelessone@boontooboiserver:~$ sudo systemctl restart apache2
```

Create the SSL Certificate

Create a directory for the certificate: **sudo mkdir /etc/ssl/private**
NOTE: The directory is created with sudo so it can only be accessed by privileged users!

Generate a **Private Key** and **Self-Signed Certificate**:
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/apache-selfsigned.key -out /etc/ssl/certs/apache-selfsigned.crt

After running the command, you will be prompted to fill out some information for your certificate:
NOTE: Make sure to put the proper FQDN of your website in the **Common Name** section!

```
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:Iowa
Locality Name (eg, city) []:Cedar Falls
Organization Name (eg, company) [Internet Widgits Pty Ltd]:UNI-SYS-ADMIN
Organizational Unit Name (eg, section) []:University of Northern Iowa
Common Name (e.g. server FQDN or YOUR name) []:slickbrickcentral.com
Email Address []:olhavad@uni.edu
```

Configure Apache to use SSL

We need to create a **VirtualHost** that will facilitate HTTPS requests on port 443.

Create a new file with: **sudo nano /etc/apache2/sites-available/slickbrickcentral.conf**

NOTE: Make sure to match the **ServerName** with the **Common Name** you chose when setting up the certificate! → **slickbrickcentral.com**

```
 GNU nano 7.2                                    /etc/apache2/sites-available/slickbrickcentral.conf *
<VirtualHost *:443>
        ServerName slickbrickcentral.com
        ServerAlias www.slickbrrickcentral.com
        DocumentRoot /var/www/slickbrickcentral.com

        SSLEngine on
        SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
        SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key
</VirtualHost>
```

Let's now make a new DocumentRoot and put an HTML file for testing purposes!

Create your website's root directory: **sudo mkdir /var/www/slickbrickcentral.com**

Create the landing page of the website:

sudo nano /var/www/slickbrickcentral.com/index.html

```
 GNU nano 7.2                                    /var/www/slickbrickcentral.com/index.html *
<h1>Test Page<h1>
```

Next we need to enable the configuration file: **sudo a2ensite slickbrickcentral.conf**

Test the configuration file with the command: **sudo apache2ctl configtest**

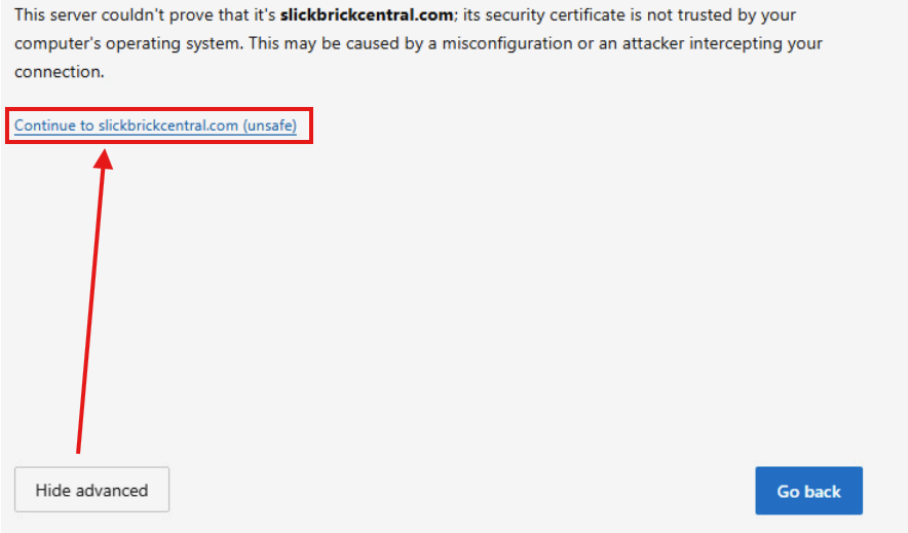
Activate changes with the command: **sudo systemctl reload apache2**

```
onelessone@boontooboiserver:/etc/ssl$ sudo a2ensite slickbrickcentral.conf
Enabling site slickbrickcentral.
To activate the new configuration, you need to run:
  systemctl reload apache2
onelessone@boontooboiserver:/etc/ssl$ sudo apache2ctl configtest
Syntax OK
onelessone@boontooboiserver:/etc/ssl$ sudo systemctl reload apache2
```

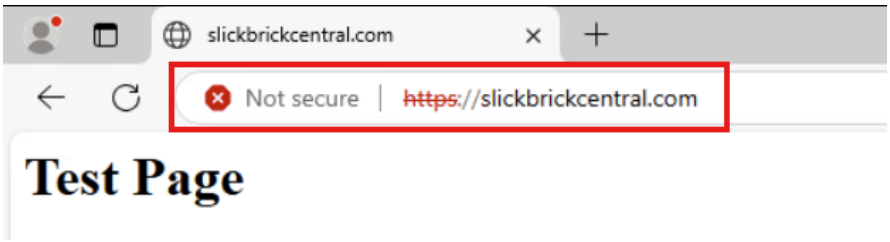
Test HTTPS Connection

Go to your DMZ Client and visit your website with HTTPS!
In a browser enter https://slickbrickcentral.com

Upon navigating to your page you will see a warning about the self-signed certificate. You can click "Advanced" towards the bottom of the warning graphic and then proceed to the website!



You then should see your website landing page render with the simple message coded earlier. Notice that the browser URL is using **HTTPS**!



Redirect HTTP to HTTPS

Currently, our configuration will only respond to HTTPS requests on port 443. It is good practice to also respond on port 80, even if you want to force all traffic to be encrypted. Let's set up another **VirtualHost** to respond to these unencrypted requests and redirect them to HTTPS.

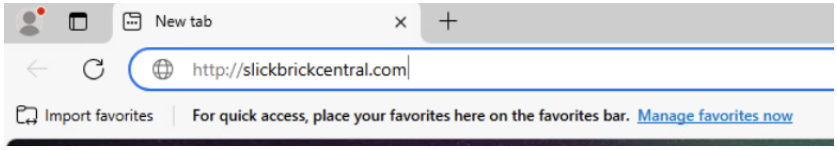
Open the same Apache configuration file:
sudo nano /etc/apache2/sites-available/slickbrickcentral.conf

```
 GNU nano 7.2                                  /etc/apache2/sites-available/slickbrickcentral.conf *
<VirtualHost *:443>
        ServerName slickbrickcentral.com
        ServerAlias www.slickbrrickcentral.com
        DocumentRoot /var/www/slickbrickcentral.com

        SSLEngine on
        SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
        SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key
</VirtualHost>

<VirtualHost *:80>
        ServerName slickbrickcentral.com
        ServerAlias www.slickbrickcentral.com
        DocumentRoot /var/www/slickbrickcentral.com
        Redirect / https://slickbrickcentral.com/
</VirtualHost>
```

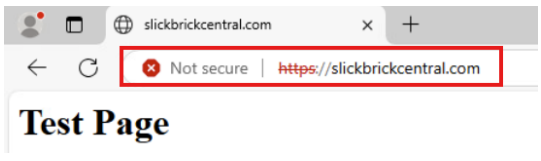
Next we need to refresh the configuration file: **sudo a2ensite slickbrickcentral.conf**
Test the configuration file with the command: **sudo apache2ctl configtest**
Activate changes with the command: **sudo systemctl reload apache2**

```
onelessone@boontooboiserver:/etc/ssl$ sudo a2ensite slickbrickcentral.conf
Site slickbrickcentral already enabled
onelessone@boontooboiserver:/etc/ssl$ sudo apache2ctl configtest
Syntax OK
onelessone@boontooboiserver:/etc/ssl$ sudo systemctl reload apache2
```

Test the redirect by trying to access your website with HTTP protocol!
You can do this by navigating in a browser to http://slickbrickcentral.com



The website should **redirect** the HTTP request to HTTPS!

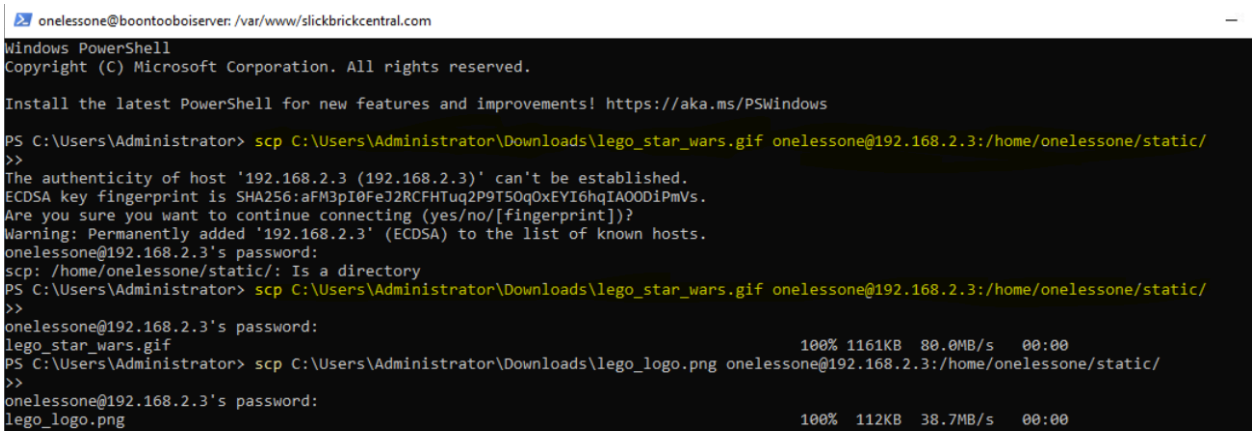


Disable the default configuration file: **sudo a2dissite 000-default.conf**
Reload Apache to apply changes: **sudo systemctl reload apache2**

Static HTML Page

After verifying the server is running on our **Ubuntu Server at 192.168.2.3**, we need to create a customized static HTML page to display!

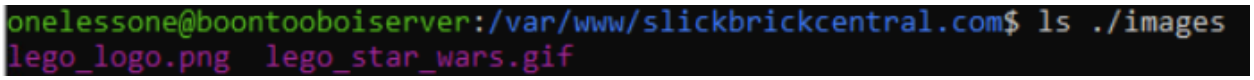
I downloaded external images to a DMZ client and then used **scp** to transfer them to the Apache Web Server:
scp <the/local/file/path> user@[ip_address]:[the/server/destination/path]



I then used ssh on the DMZ client to connect to the Apache server:
ssh onelessone@192.168.2.3

After logging in, I created an images directory within the website's root directory:
sudo mkdir /var/www/slickbrickcentral.com/images

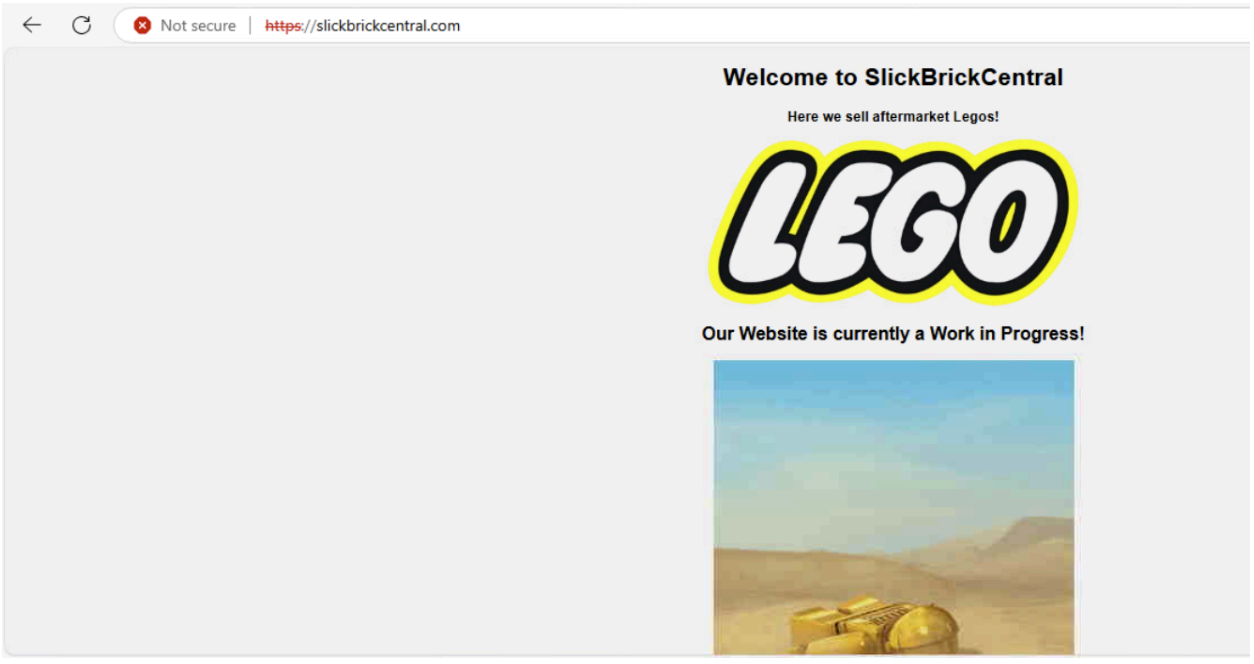
I then copied the uploaded files within the /static directory and transferred them to /images:
sudo cp /home/onelessone/static/* /var/www/slickbrickcentral.com/images/



After copying my website resources, I edited the base HTML page:
sudo nano /var/www/slickbrickcentral.com/index.html

```
onelessone@boontooboiserver: ~
 GNU nano 7.2                                  /var/www/slickbrickcentral.com/index.html
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to SlickBrickCentral</title>
    <style>
        body {
            text-align: center;
            font-family: Arial, sans-serif;
            background-color: #f0f0f0;
        }
        img {
            max-width: 100%;
            height: auto;
        }
    </style>
</head>
<body>
    <h1>Welcome to SlickBrickCentral</h1>
    <h3> Here we sell aftermarket Legos!<h3>
    <img src="images/lego_logo.png" alt="LEGO Logo">
    <h2>Our Website is currently a Work in Progress!</h2>
    <img src="images/lego_star_wars.gif" alt="Lego Star Wars GIF">
</body>
</html>
```

After saving the file, render the new page contents in a browser: https://slickbrickcentral.com



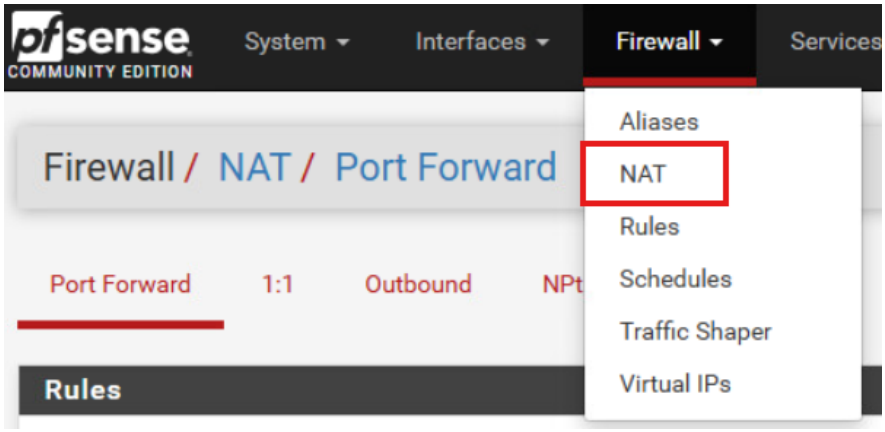
pfSense Integration

I will have to **port forward** http and https traffic entering my pfSense address from the WAN to the internal Apache Web Server.

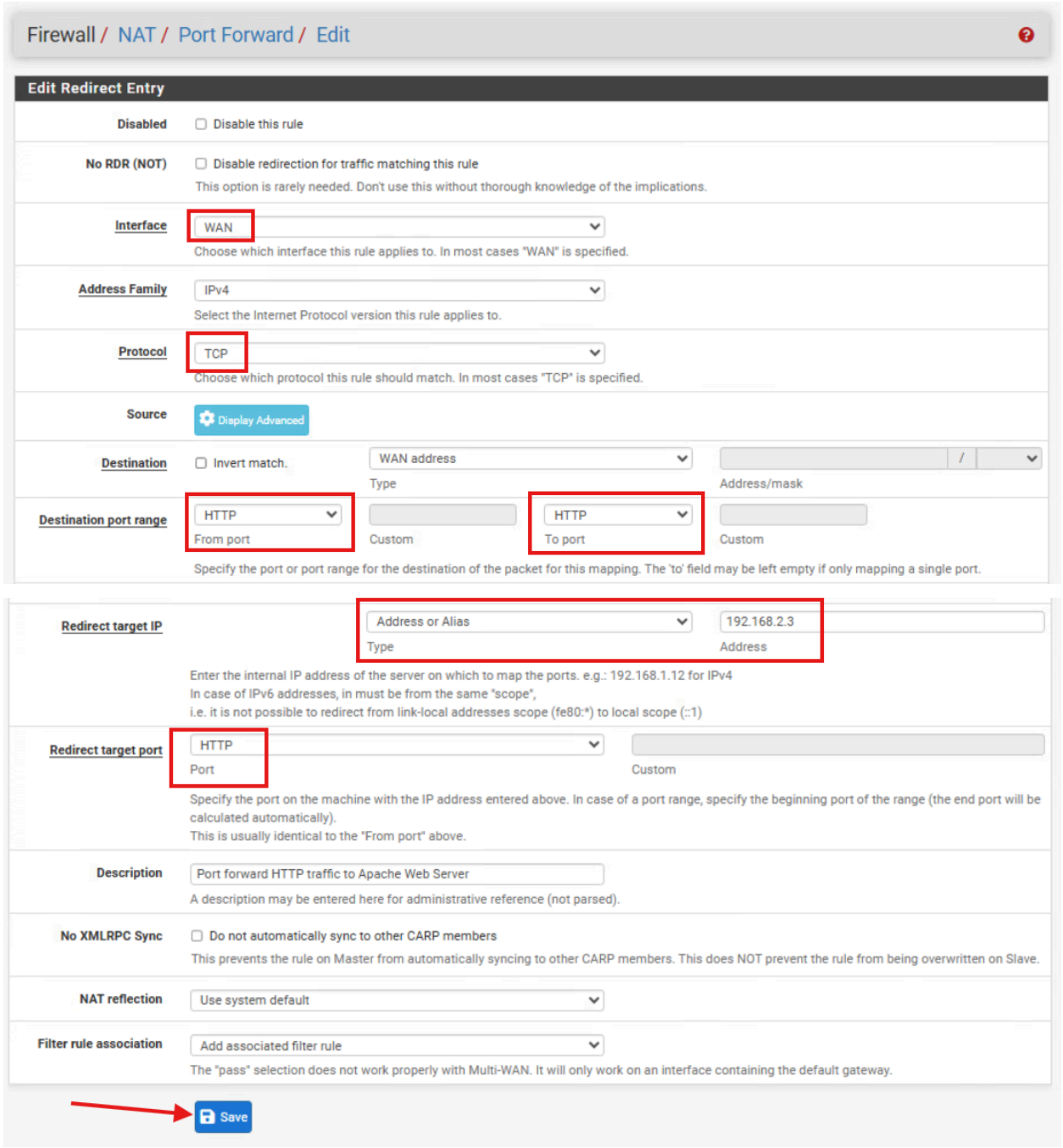
To do this I can create a port forward rule to forward traffic on ports 80 (http) and 443 (https) from 10.161.21.4 (pfSense WAN) to 192.168.2.3 (Apache Web Server).

First, go onto a LAN machine and navigate to the pfSense configuration portal: http://192.168.1.1

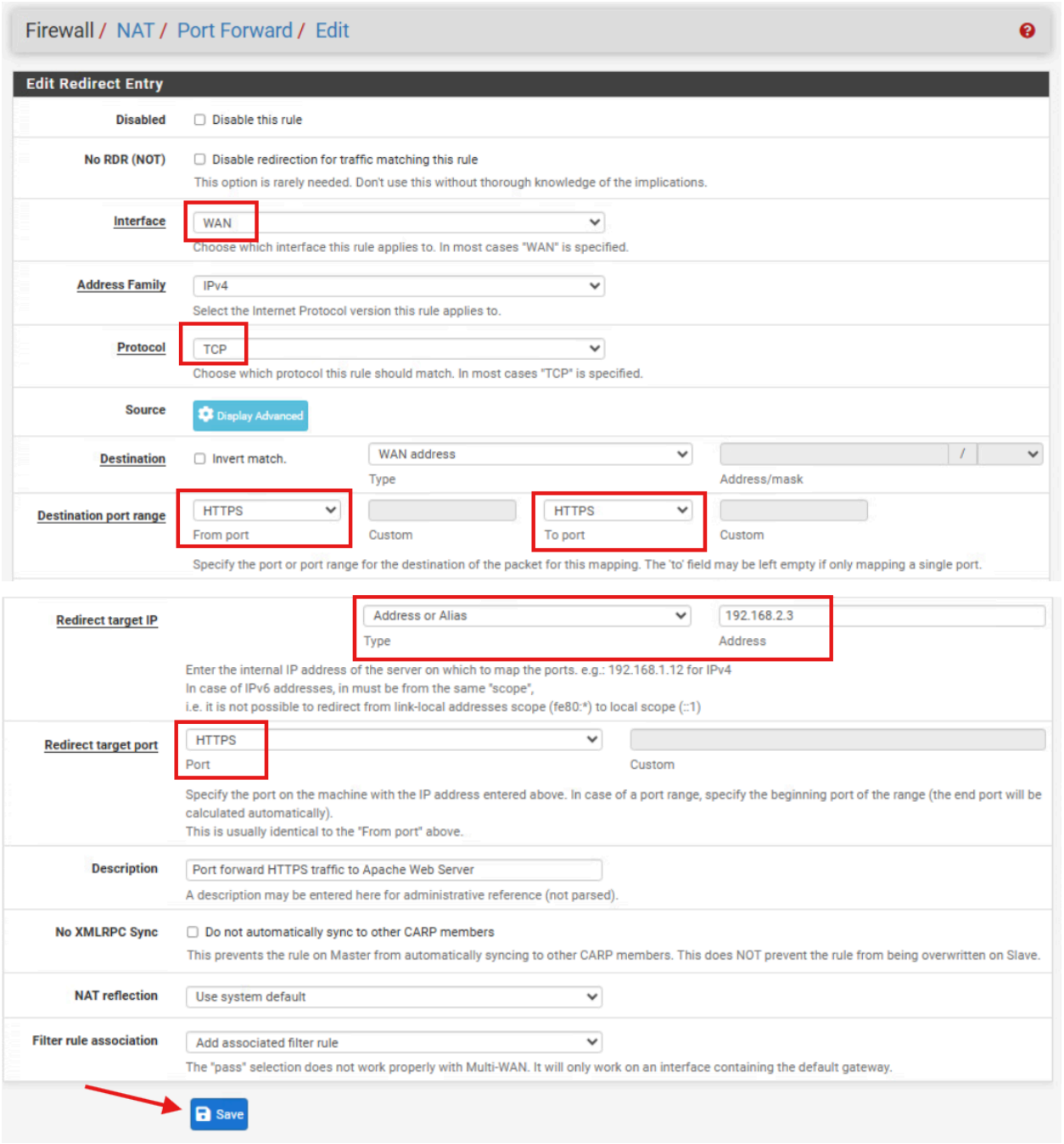
In the top bar, navigate to **Firewall > NAT > Port Forward**



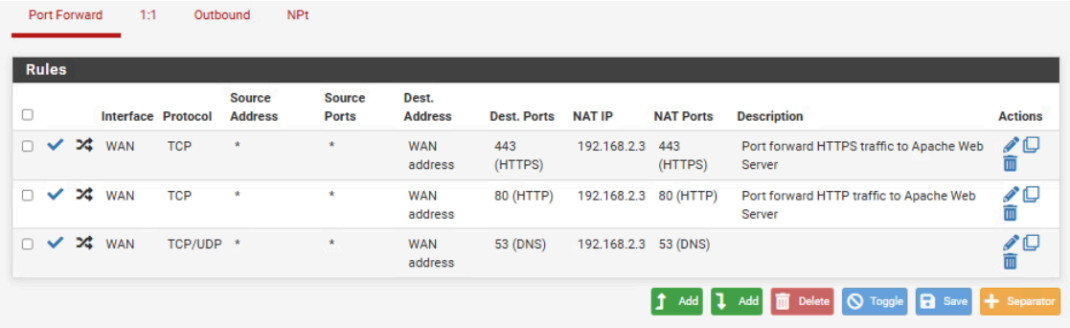
Add a new rule for **HTTP** traffic (**port 80**)



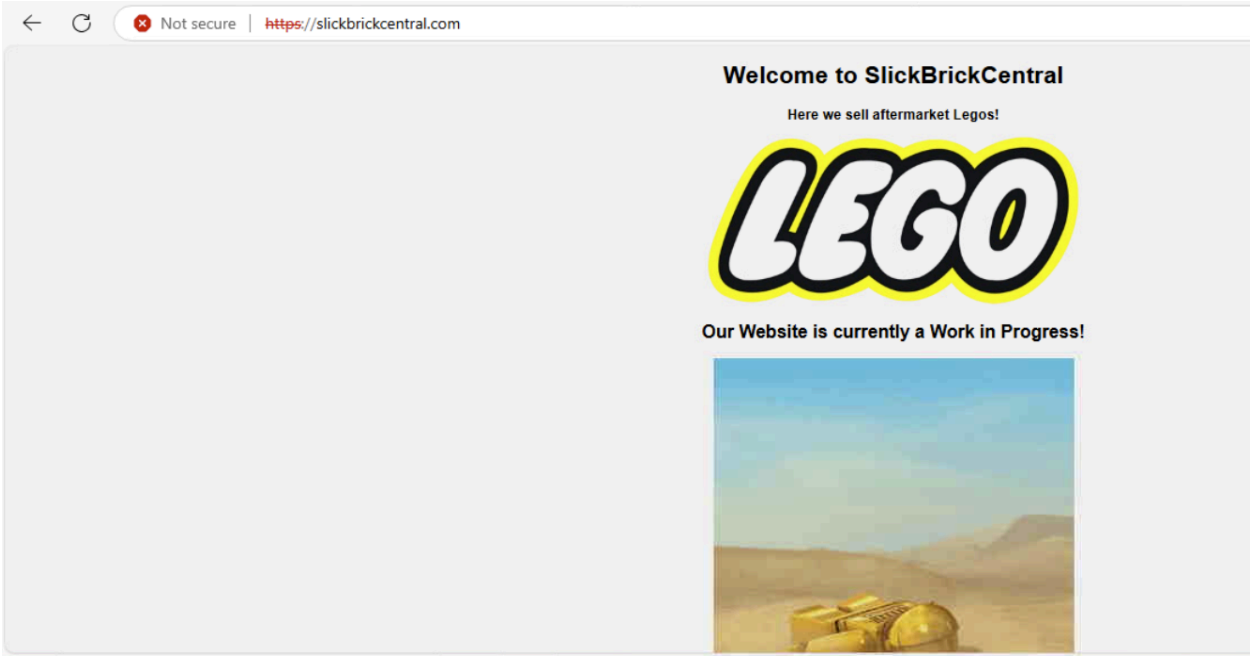
Add a new rule for **HTTPS** traffic (**port 443)**



Final Port Forward Rules (make sure to **Apply Changes**)



Go to your WAN machine and verify you can access your Apache Web Server.
In the browser I will check by using my chosen domain name: **slickbrickcentral.com**



Project 7: Active Directory

Below contains steps outlining the configuration of an **Active Directory Domain**. The **Windows 2022 Server** (**192.168.2.2**) in the DMZ will be designated as the **Domain Controller**. Active Directory **Roaming Profiles** will enable my employees to log into any LAN workstation and maintain their user files and settings.

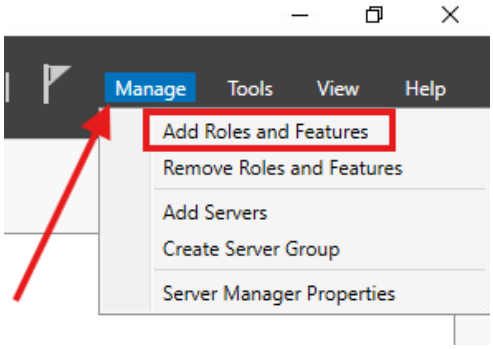
AD Information
AD Domain Controller: **192.168.2.2**
AD Domain: **ad.local**
Usernames: AD\Administrator, bill, fernando
Employee Password: **Classified**

AD Setup

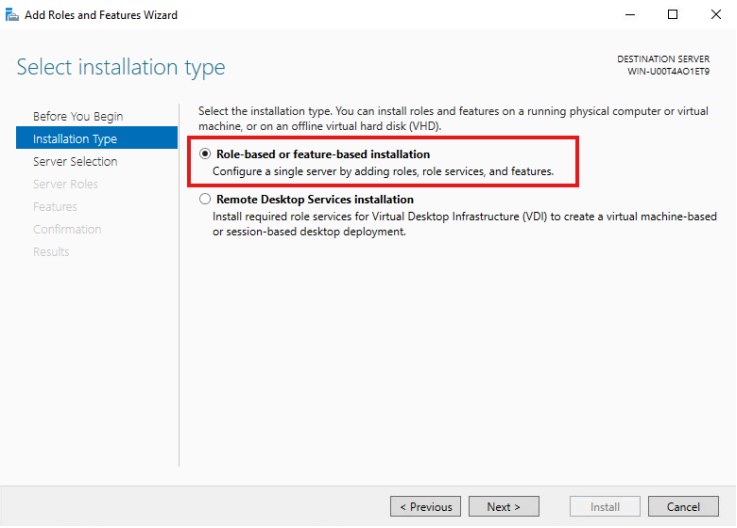
Active Directory is a Microsoft tool that helps manage users and computers in a network. It organizes and controls access to resources like files and applications. I will be installing Active Directory on my Windows 2022 Server within the DMZ under the IP Address 192.168.2.2!

Installing AD Role and Features

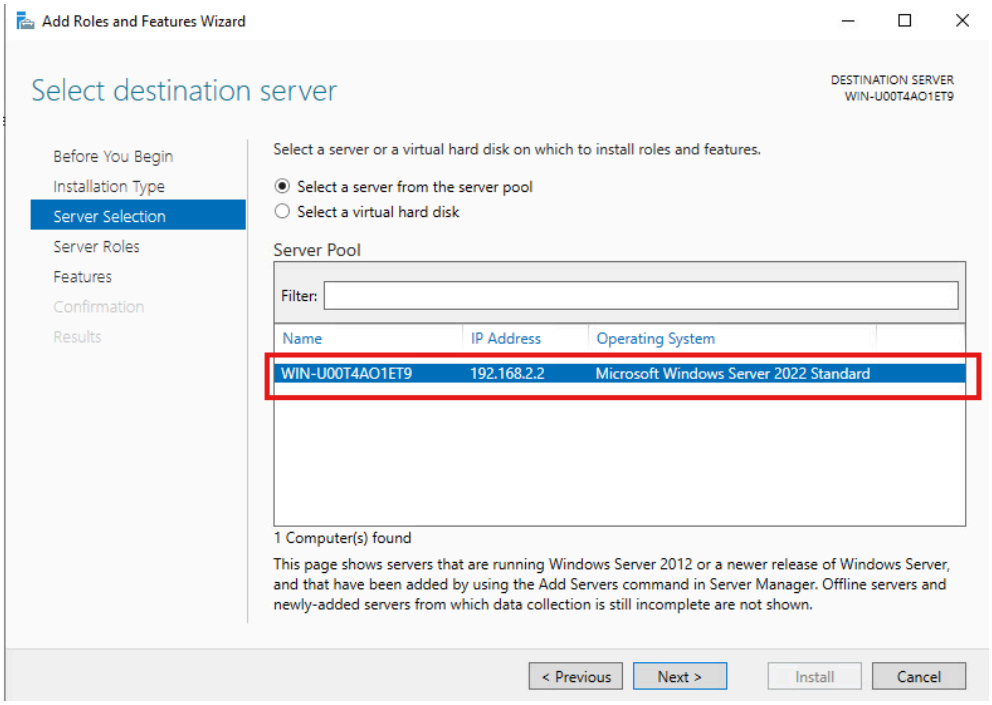
After logging into the Windows 2022 Server, open the Server Manager application.
Then in the upper right corner navigate to **Manage > Add Roles and Features**



Installation Type: Role-based or feature-based installation.

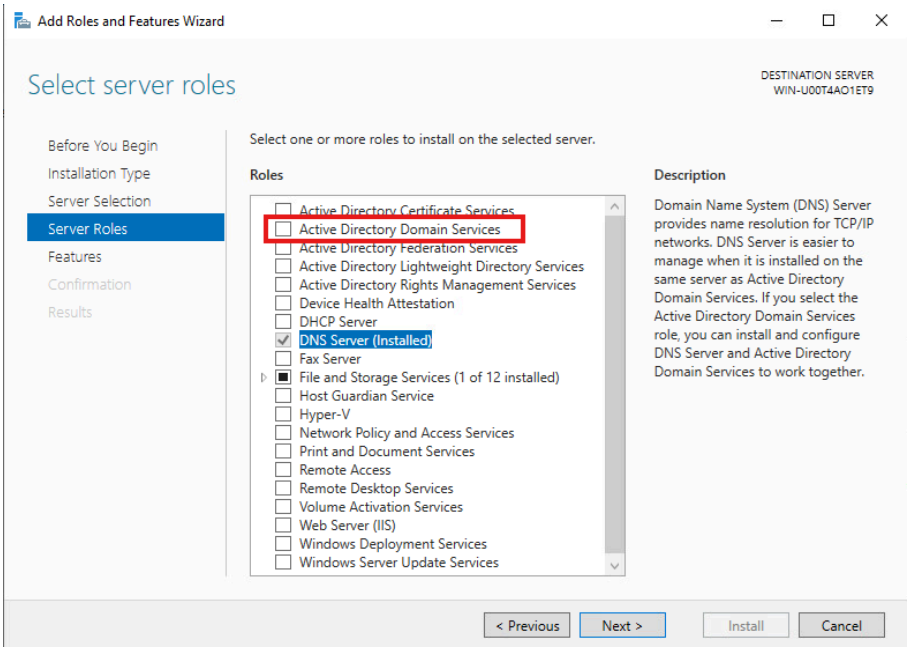


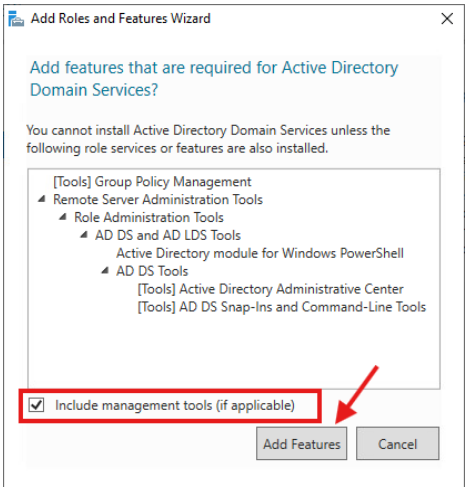
Server Selection: Select your local server where you wish to install the AD DS role. This menu shows our local server, WIN-U00T4AO1ET9, its IP and OS.



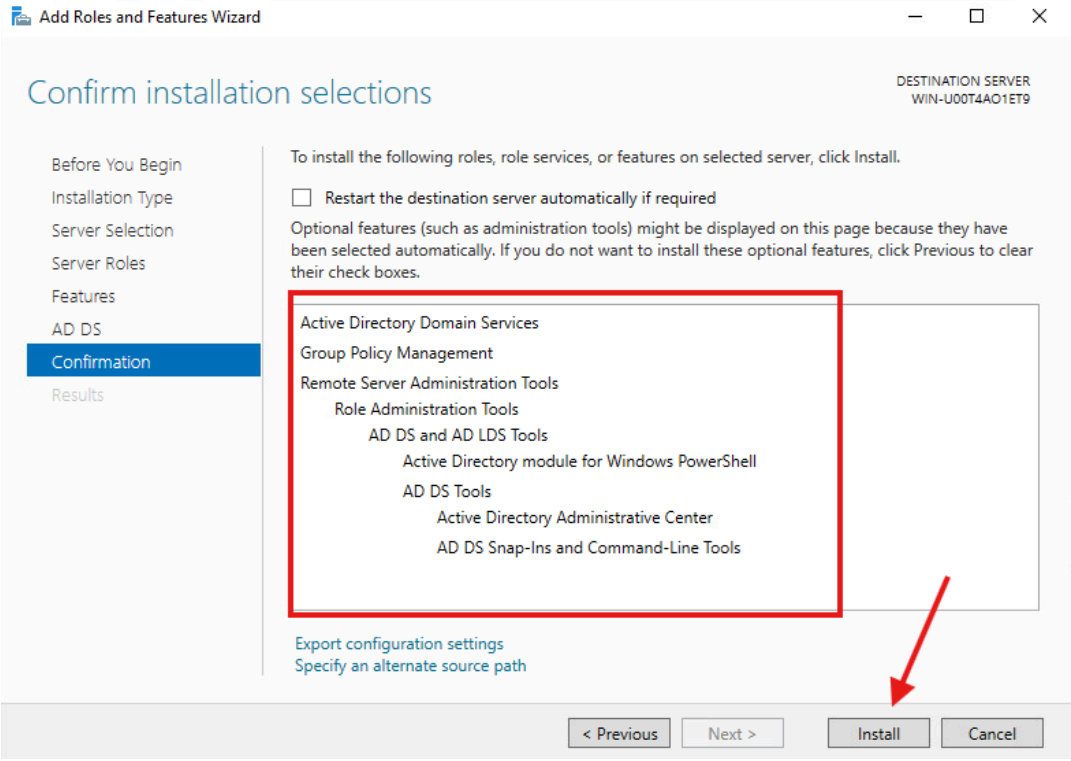
Server Roles: Select "Active Directory Domain Services". Once you choose that option, a new pop-up window will appear. This window shows the features required for AD DS that you will need to include. Select "Include management tools" and click on "Add Features".

NOTE: Disregard the DNS selection. That is not needed for Active Directory!

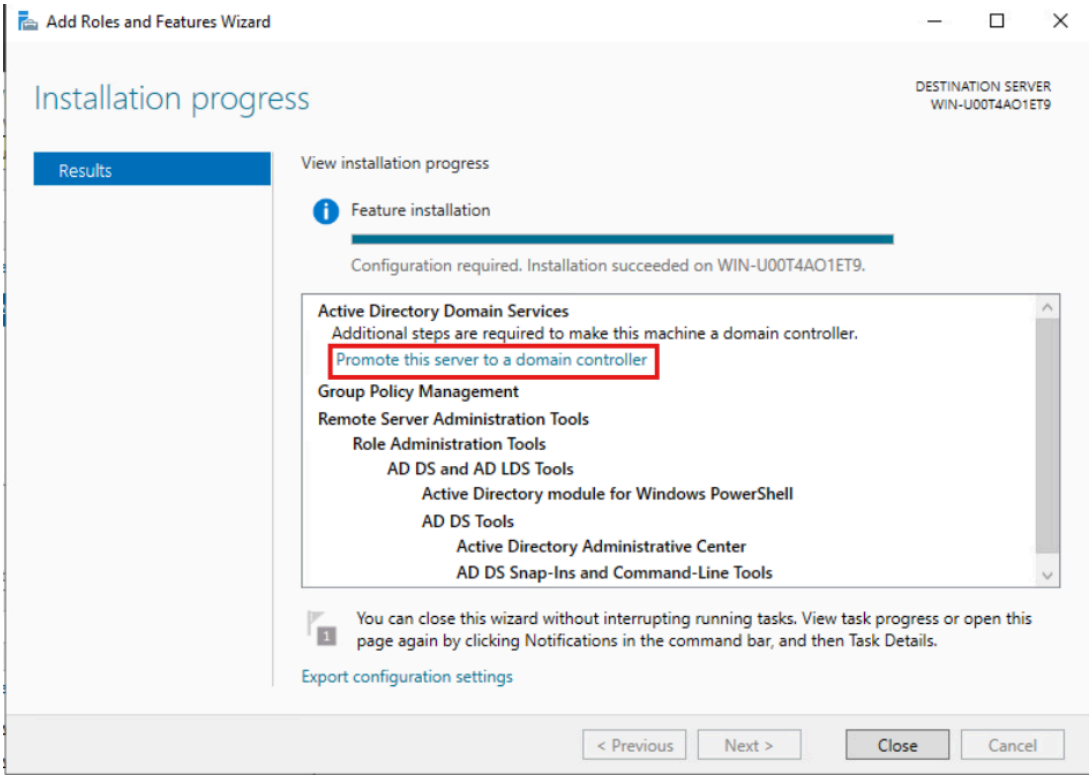




Features: Select "Next"
AD DS: Select "Next"
Confirmation: Verify the Active Directory configuration and select "Install"



NOTE: Keep the installation wizard **OPEN** until the installation is complete! Then select "Promote this server to a domain controller"

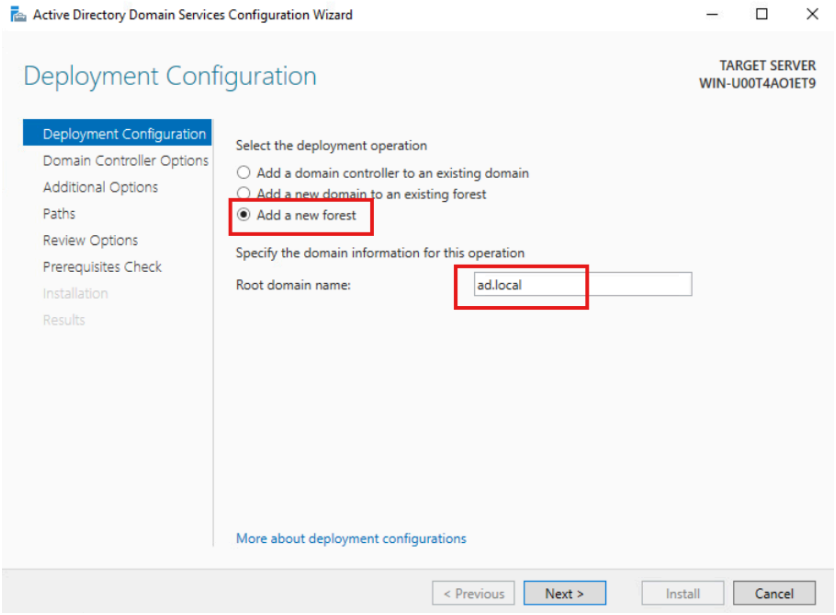


A new setup window will appear. This contains three options for deployment: “Add a domain controller to an existing domain”, “Add a new domain to an existing forest”, or “Add a new forest”. Since we are starting from scratch, we are going to create a brand new forest.

AD Forest and Domain Controller

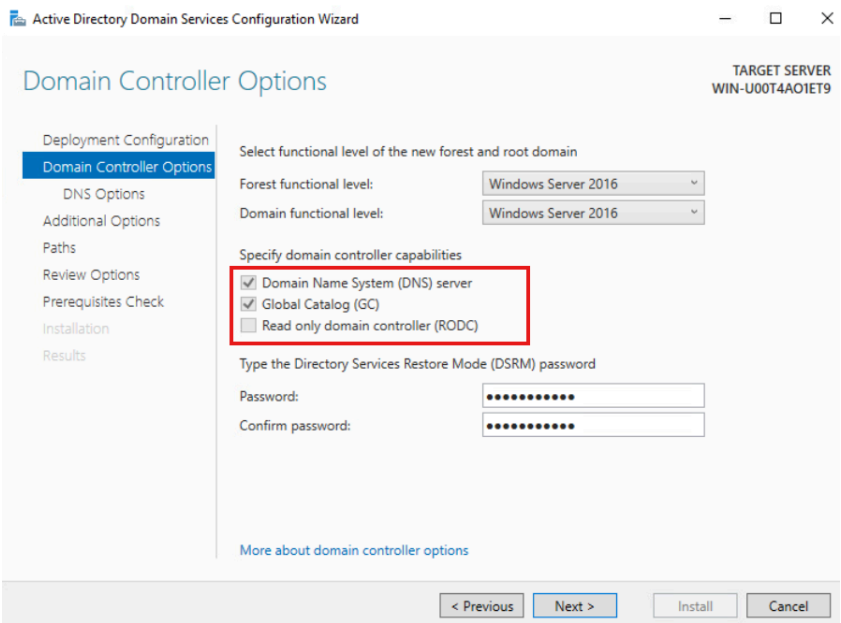
Deployment Configuration: Select “Add a new forest” and specify the Root domain name.

Root domain name: **ad.local**

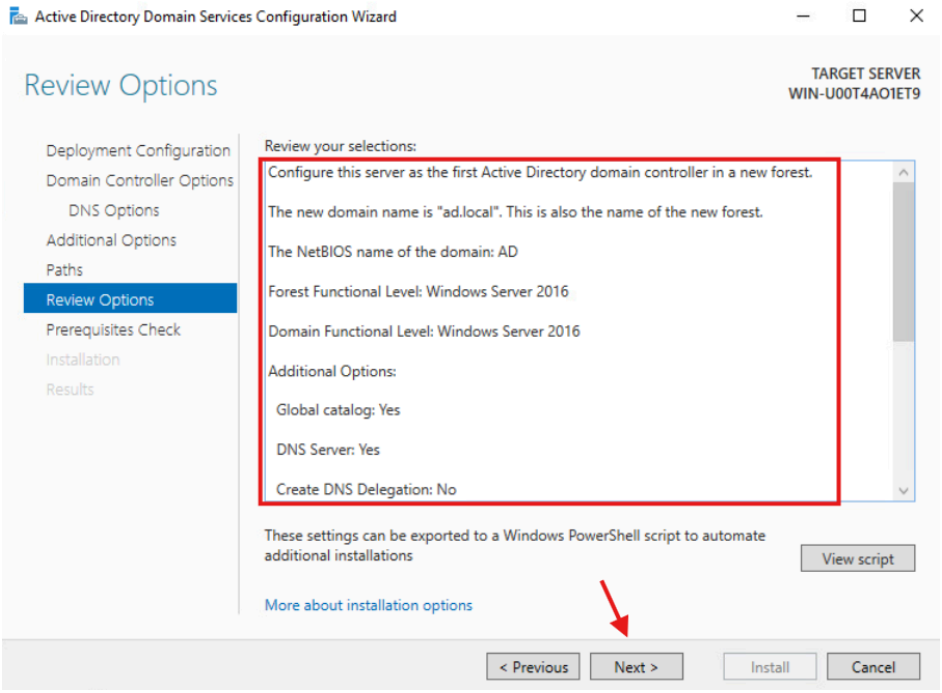


Domain Controller Options: Since this is the first AD domain controller, check the “DNS server” and “Global Catalog” boxes. Give the Directory Services Restore Mode (DSRM) a password. Then click on “Next”.

DSRM Password: **Classified**

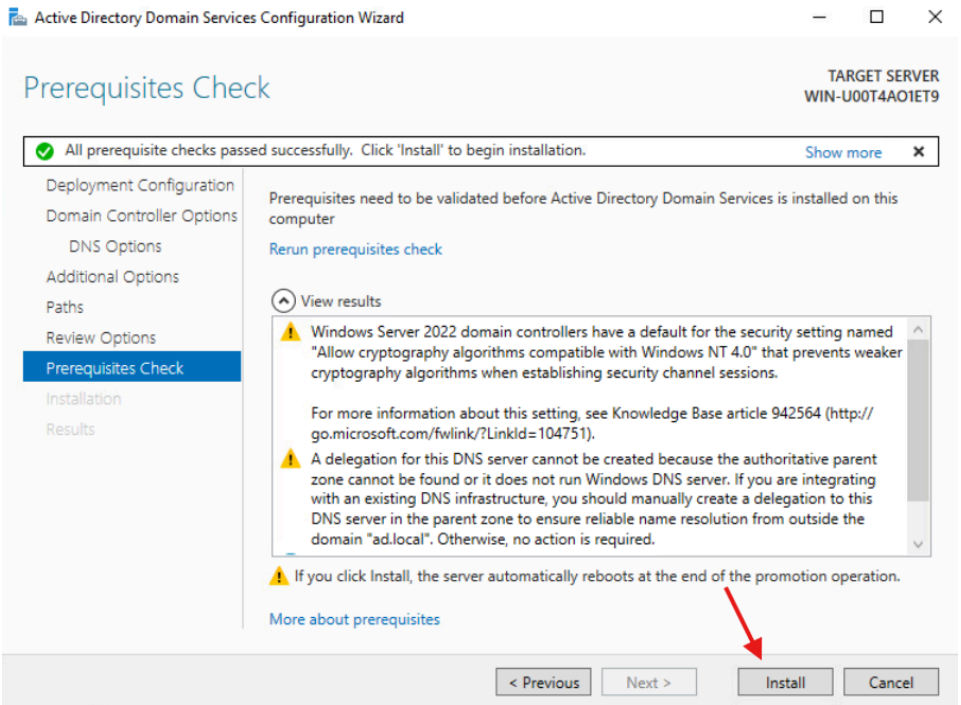


DNS Options: SKIP, no need for DNS Delegation right now
Additional Options: Displays the NetBIOS domain name taken from the root domain. Ex: AD
Paths: You can specify the location of the AD DS database, log files, and SYSVOL folders.
Review Options: Check your configuration and go back if you want to make any changes before installation. If everything looks good, click "Next".



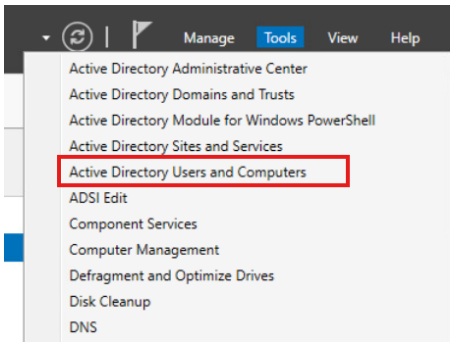
Prerequisites Check: Shows you a checklist with warnings or critical alarms. If the check passes, it will show a green light, and you'll be able to install AD DS with the new domain controller and forest. You can take care of these warnings later on.

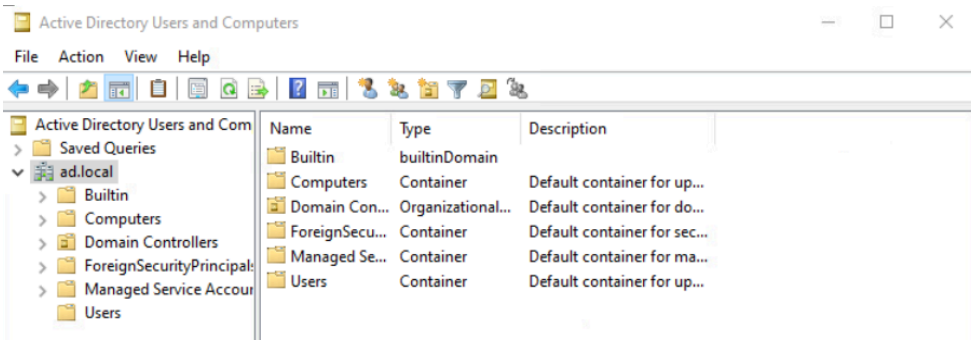
Click "Install". The server will then install and **restart** automatically.

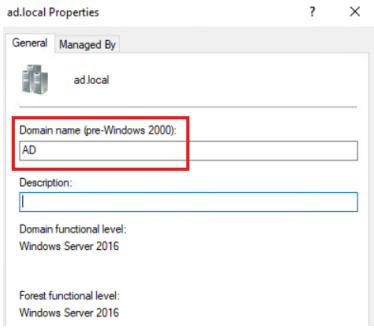


After the server restarts, verify your configuration by opening the Server Manager application.

Then in the upper right corner navigate to **Tools > Active Directory Users and Computers**. You should see the domain name you picked! My domain was **ad.local**.







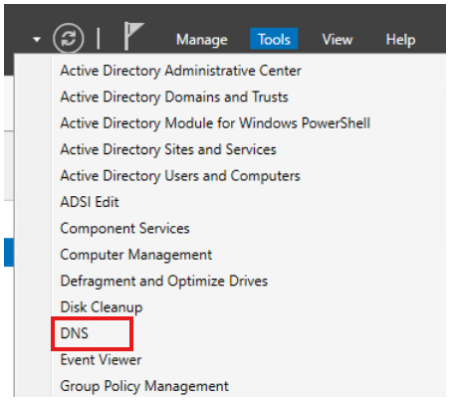
AD DNS Server Zones

You already installed AD DS, the DNS role, and created a new Forest and Domain Controller (DC). Now you only need to configure the DNS zones.

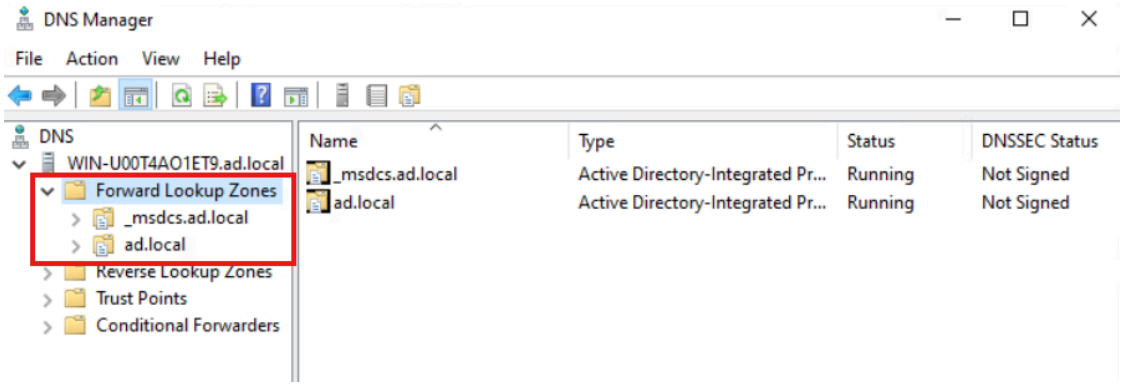
A DNS zone is formed by the mappings of IPs and hostnames used to resolve DNS queries. The most common zone type in Active Directory is the **Active Directory-integrated DNS zone**.

Forward Lookup Zone

In the upper right corner navigate to **Tools > DNS**

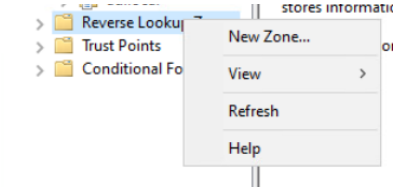


Expand the "Forward Lookup Zones" directory and you will see the root domain name chosen earlier (**ad.local**) and the **_msdcs** zone which is created by default.

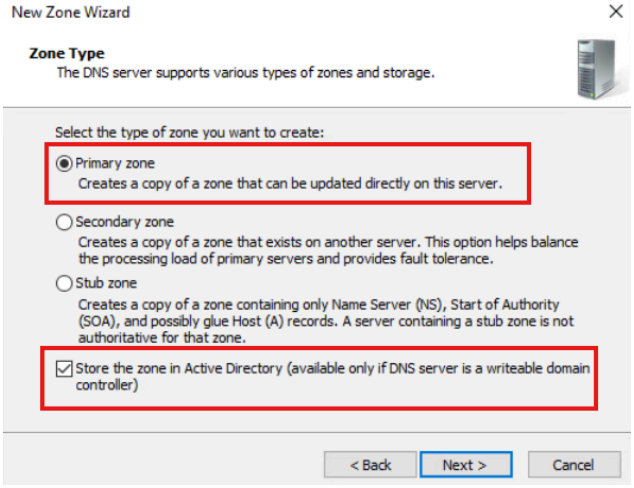


Reverse Lookup Zone

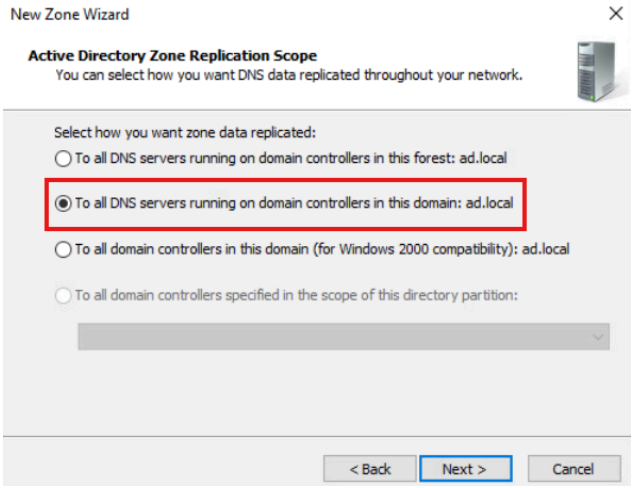
The Forward Lookup Zone is already pre-populated but the Reverse Lookup Zone is not. Create a new zone by right clicking the "Reverse Lookup Zones" directory and selecting "New Zone".



Zone Type: Check "Primary Zone" and "Store the zone in Active Directory"



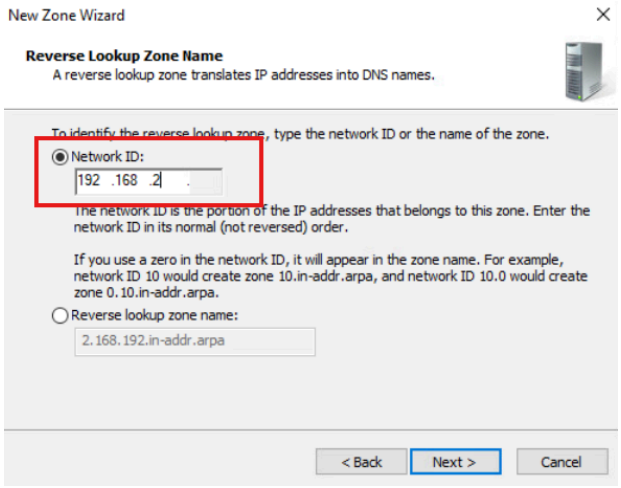
AD Zone Replication Scope: Check "To all DNS servers running on domain controllers in this domain: <your domain>"



Reverse Lookup Zone Name: Select "IPv4 reverse lookup zone" and click "Next"

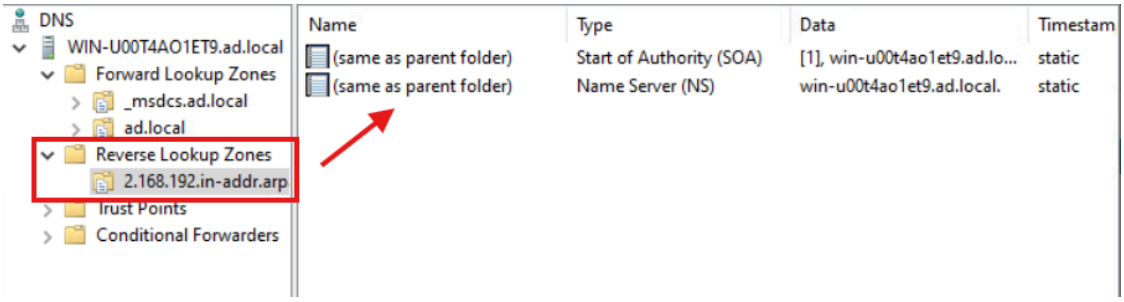
Reverse Lookup Zone Name: Specify the **Network ID** of the reverse zone to help identify the reverse lookup zone. The **first three octets** of your Windows Server's IP are used for the Network ID.

Windows 2022 Server: **192.168.2**.2



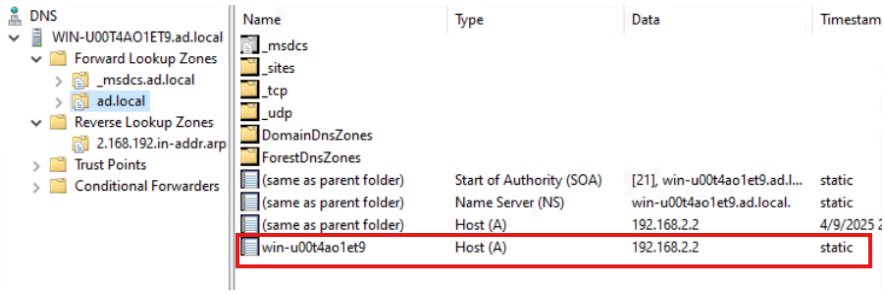
Then, click "Next > "Next" > "Finish".

Check your new Reverse Lookup Zone. Inside, you should see two DNS resource records, **SOA** and **NS**.

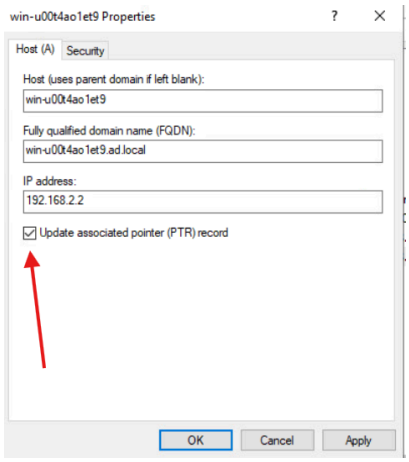


Update A Records

Open your domain name directory within the Forward Lookup Zone and then double-click the Host A record.

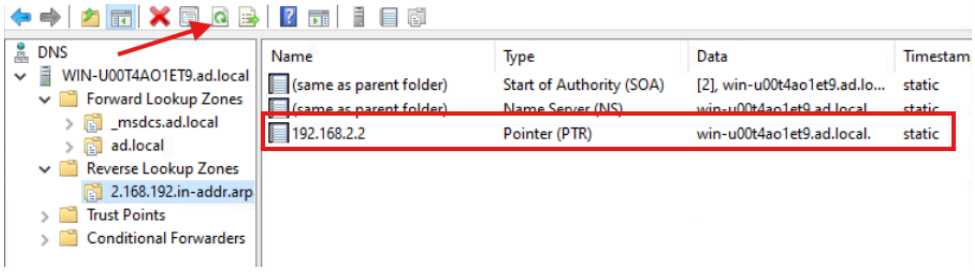


A popup window will open and make sure to select "Update associated pointer (PTR) record". Then click on "Apply" and "Ok" to save changes.



Verify Reverse Lookup Zone

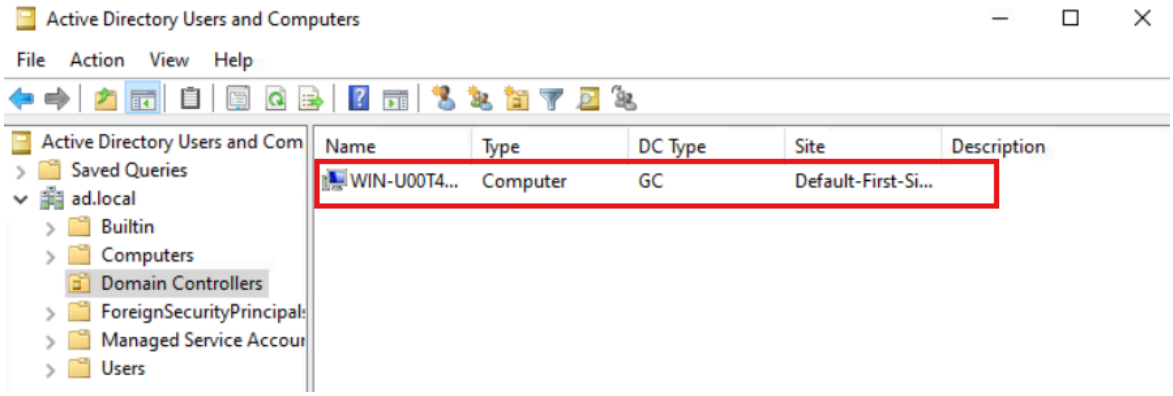
If you don't see the PTR record in the reverse lookup zone, you'll need to refresh. Click the refresh button on the top bar and the new PTR record should appear.



AD Domain Controller Verification

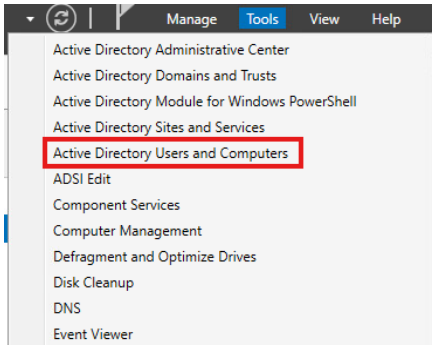
Verify your server is the AD Controller by navigating in the top right corner of Server Manager to **Tools > Active Directory Users and Computers**.

Once the pop-up opens, expand your domain name and then look into the "Domain Controllers" section. Your current machine should be listed!

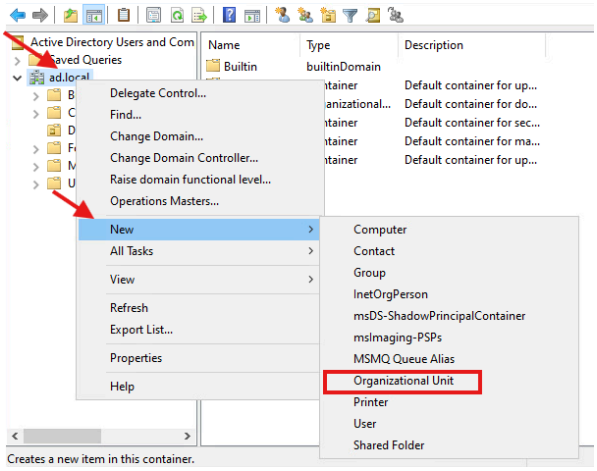


Creating AD Users

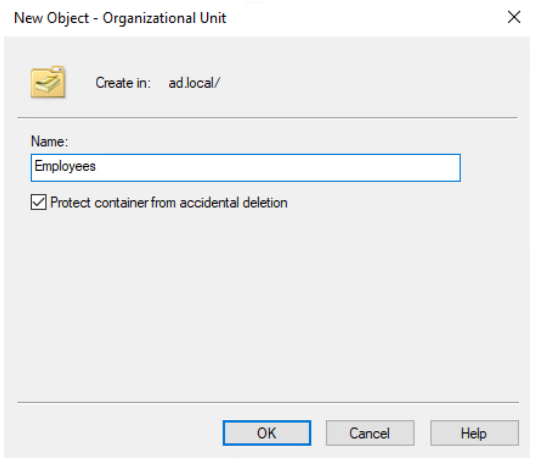
In order to create users for your Active Directory open the Server Manager application. Next, in the upper right hand corner navigate to **Tools > Active Directory Users and Computers**.



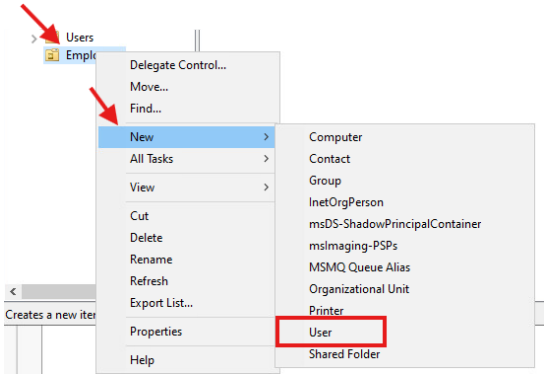
A pop-up window will appear. Right-click on your domain and then navigate to **New > Organizational Unit**.



Name your Organizational Unit. I will be naming mine ***Employees*** since it will contain worker data.

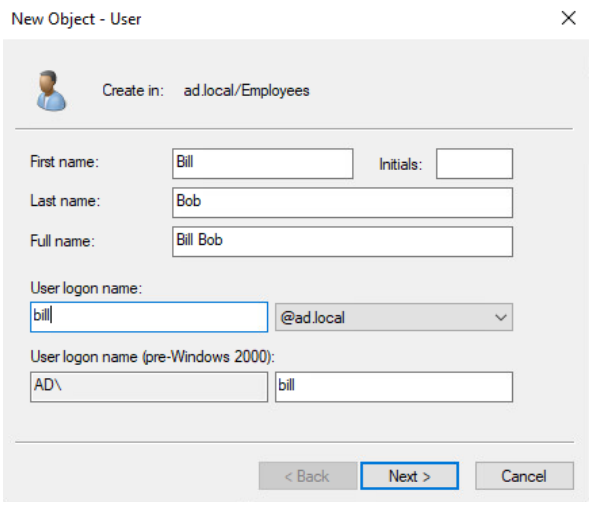


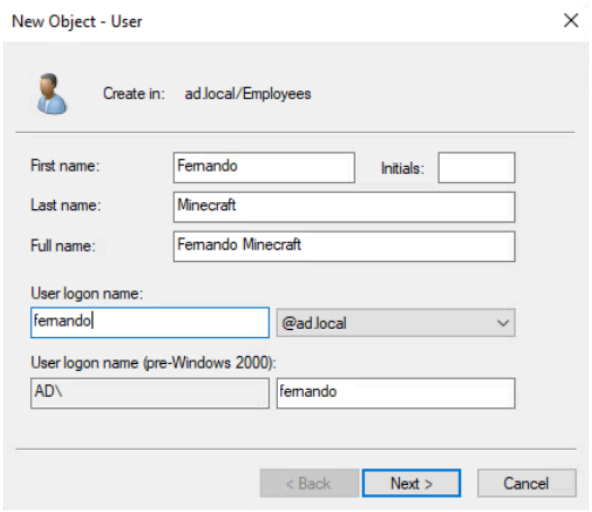
Right-click the newly created OU and then navigate to **New > User**.

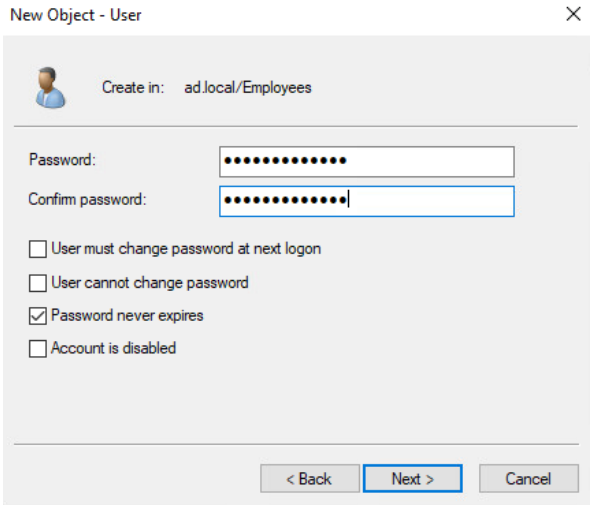


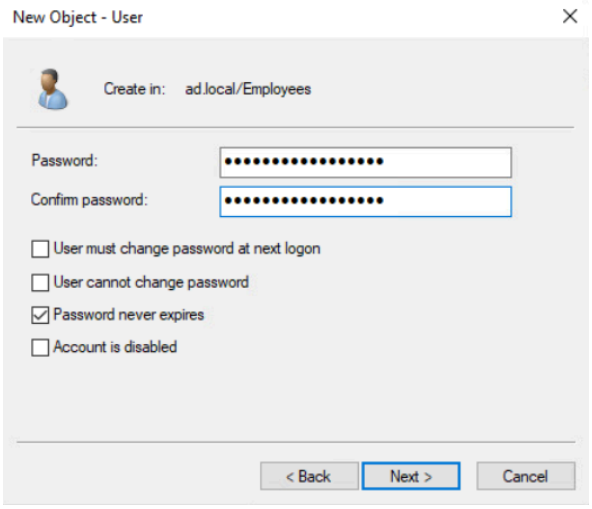
I will be creating two users for my workforce: **Bill** and **Fernando**
Note: Make sure to check the box "Password never expires" so you will not have to change it on the first logon for your users.

Usernames: **bill** or **fernando**
Password: **Classified** (same for both users)



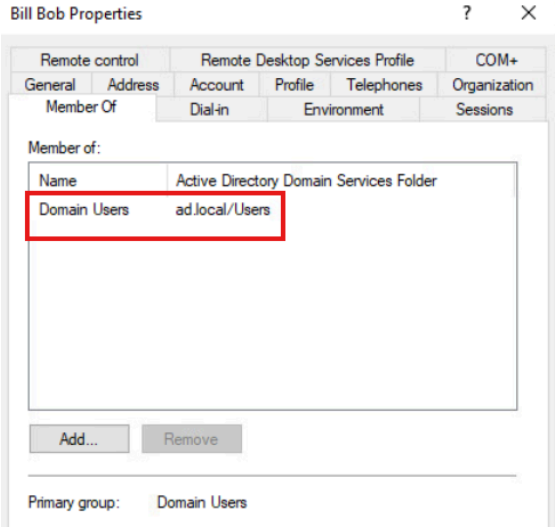






After creating both users you should see them under the ***Employees*** OU!

To validate they are in your Active Directory Domain, right-click each user and then select "Properties". Then navigate to "Member Of" and verify your domain is listed.



Success! Next, we'll join two workstations to our AD domain, allowing us to sign in with the users we've just created.

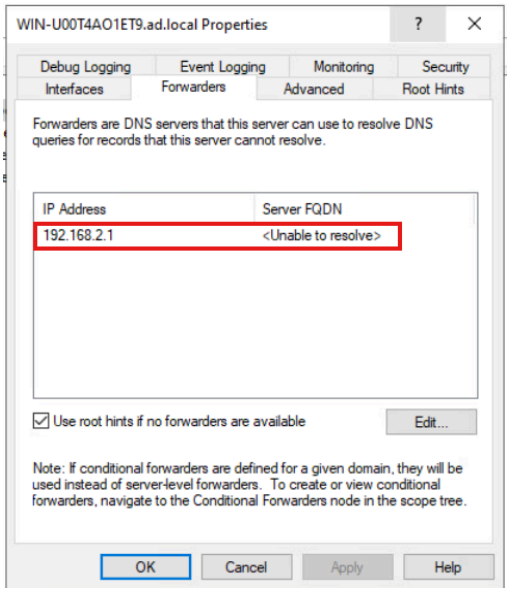
Joining Clients to AD

In order to add a windows workstation to an Active Directory, it must be able to connect to a network with an AD Domain Controller. Luckily, our LAN workstations can communicate with DMZ hosts like the AD Domain Controller!

Client DNS Setup

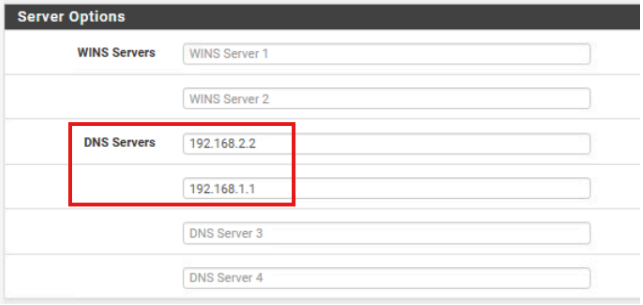
In the Windows 2022 Server, open the Server Manager application, navigate to the top right corner and select **Tools > DNS**.

A DNS Manager pop-up will appear. Within the window, right-click the DNS server name and select “Properties”. Navigate to the Forwarders section, input your preferred DNS server forwarder, and click “Apply”. I will be choosing to forward to my pfSense server at **192.168.2.1**.



This will ensure all unknown queries given to the AD Domain Controller will be forwarded elsewhere to resolve.

Next, open the pfSense Admin portal in a browser at http://192.168.1.1 and login. Once in the portal navigate in the top bar to **Services > DHCP Server**. Make sure you are in the "LAN" tab and then scroll until you see DNS servers and assign the AD Domain Controller as primary DNS (**192.168.2.2**) and then pfSense as secondary DNS (**192.168.1.1**).

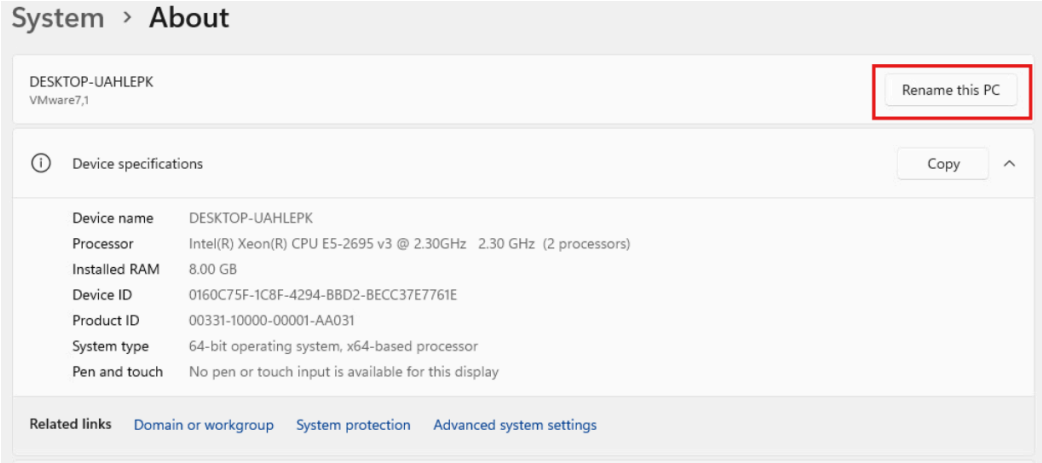


Doing this will allow all LAN workstations to be automatically issued the AD Domain Controller for DNS: **ad.local**.

Renaming Client Device

On your Windows workstation open the Settings app. Then navigate to **System > About**. Click "Rename this PC" in the upper right hand corner and create a unique name to the two client computers.

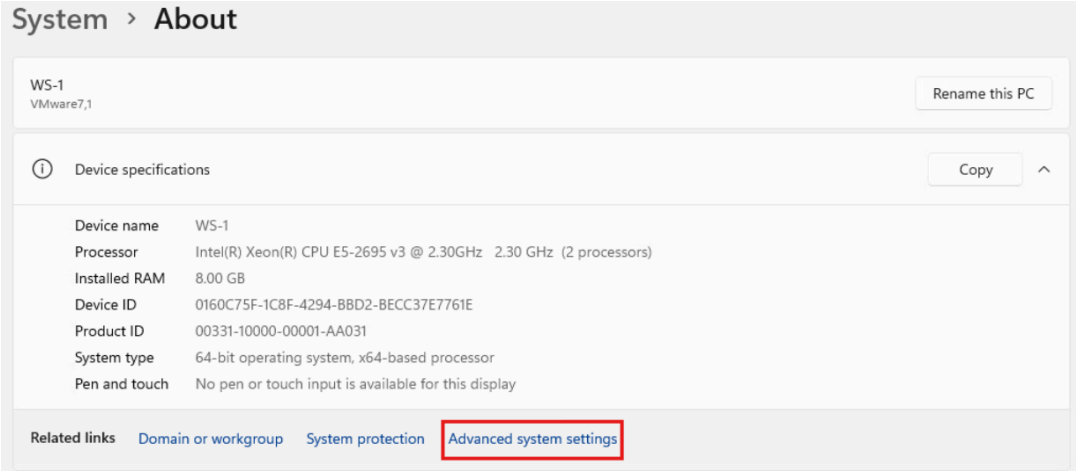
Workstation Names: **WS-1** and **WS-2**



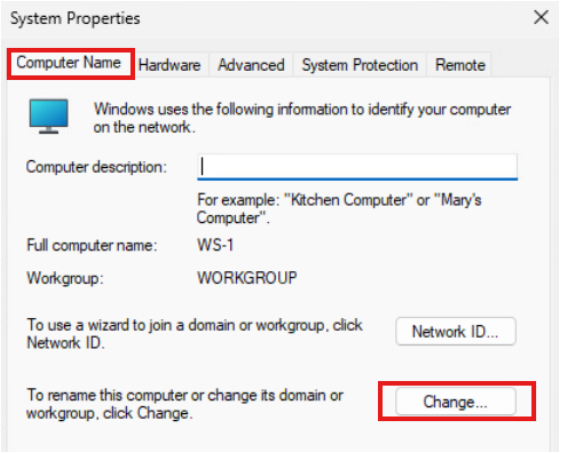
After renaming each workstation, restart the device!

Connecting Client to AD

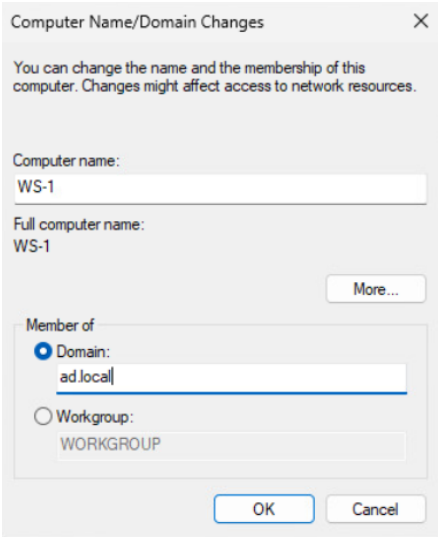
Open the Settings app again and navigate to **System > About** menu, and then select "Advanced system settings".



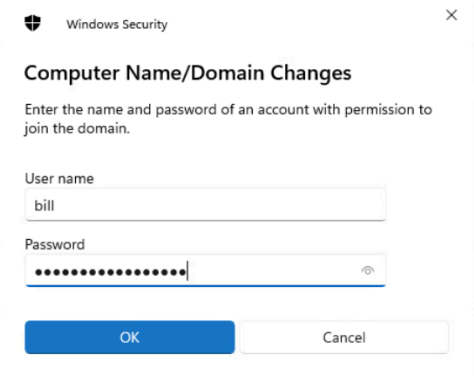
A pop-up window will open. Navigate to "Computer Name" on the top bar and click the "Change" button.



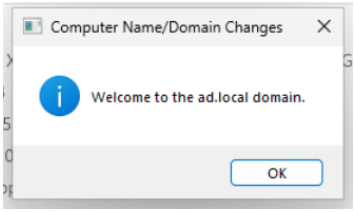
You then need to enter your domain name: **ad.local** and then click "Ok".



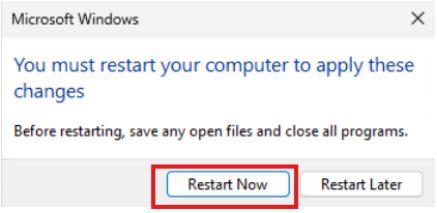
Enter the credentials of one of the AD users configured earlier to join the domain!
Username: **bill** or **fernando**
Password: **Classified**



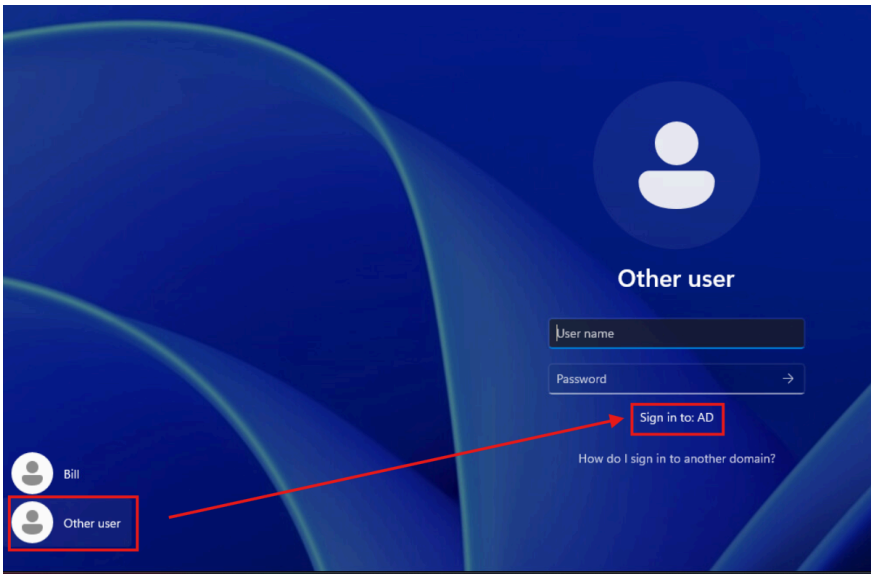
If login is a success, you will see the following message.



Restart to apply changes!



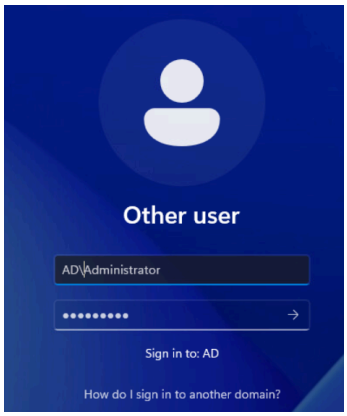
After restart, select "Other user" login and you should see your AD domain listed. You then can sign in with one of the AD user accounts.



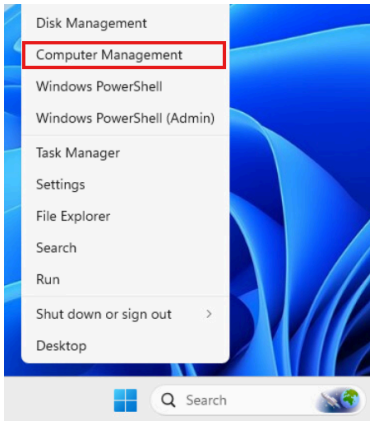
Delete Local Accounts

There is no need for the local account on our Windows workstation since we have Active Directory connected. Let's delete it!

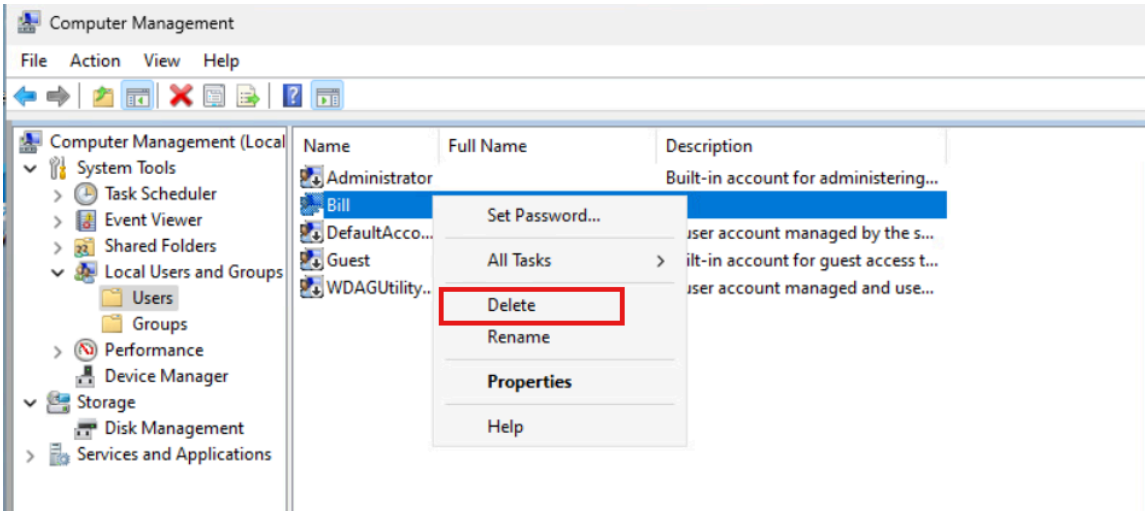
Login to your workstation with the Administrator Active Directory user.
Username: **AD\Administrator**



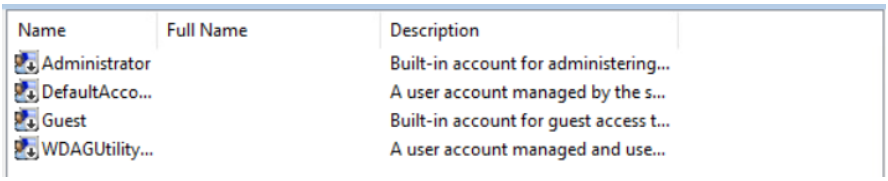
Right-click the Windows button and select "Computer Management".



Navigate into Local Users and **Groups > Users** and right-click the local user you want to delete.



After deletion, there should be no local users remaining!

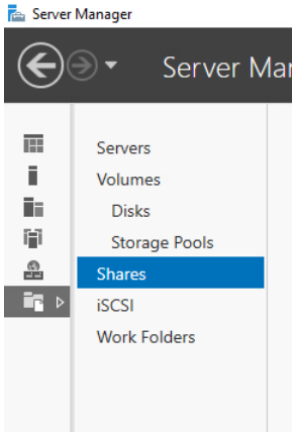


Name	Full Name	Description
Administrator		Built-in account for administering...
DefaultAcco...		A user account managed by the s...
Guest		Built-in account for guest access t...
WDAGUtility...		A user account managed and use...

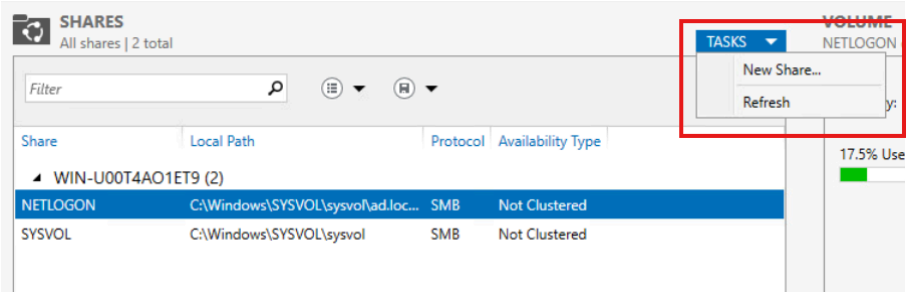
Roaming Profiles

Active Directory **Roaming Profiles** are a Windows Server feature that allow users to access their personal desktop settings and files from any computer within the domain.

Before configuring a Roaming Profile, we need to create a Share. Open the Server Manager application and navigate to **File and Storage Services > Shares**.

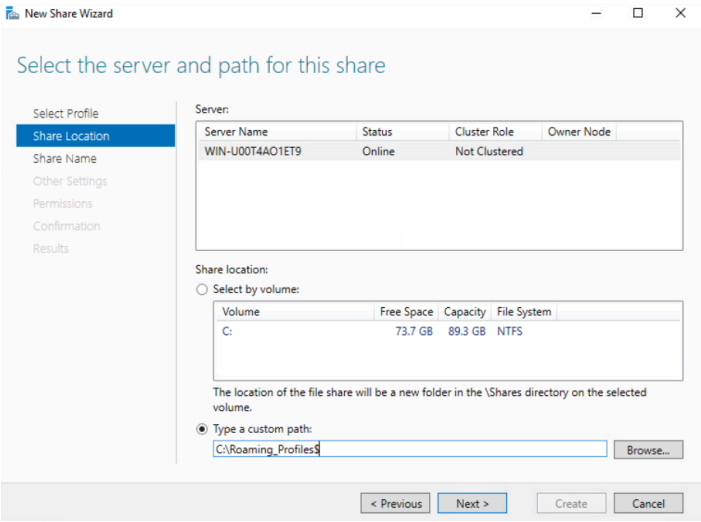


Create a new Share.



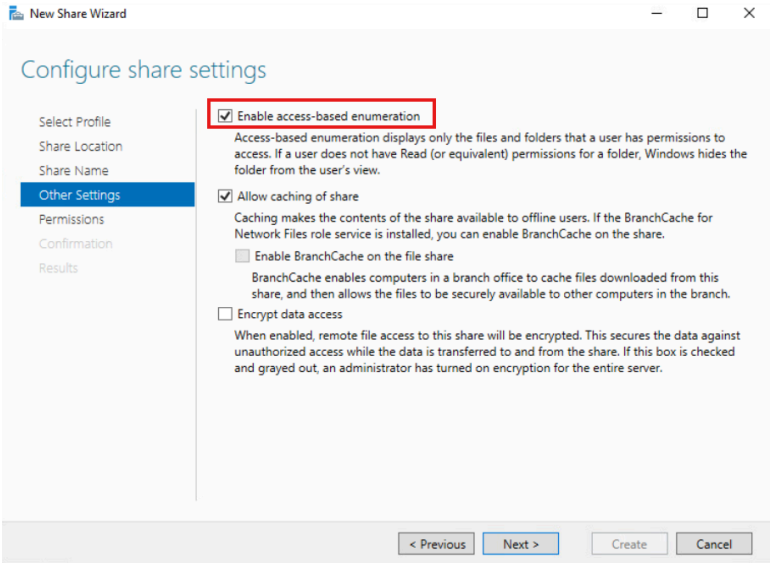
Select Profile: SMB Share - Quick

Share Location: C:\Roaming_Profiles$

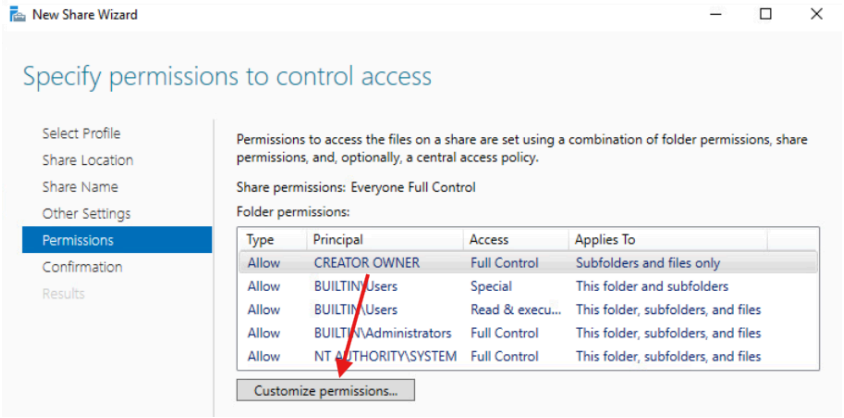


Share Name: Click “Next”

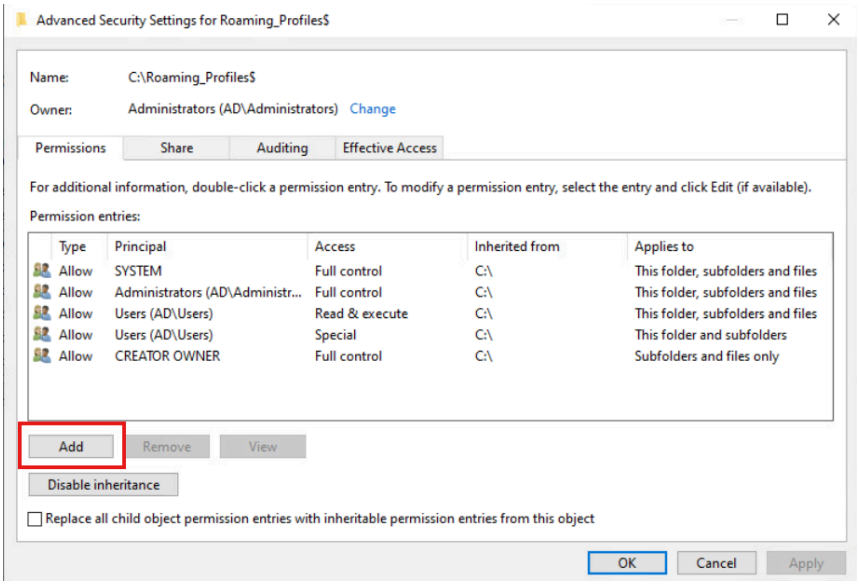
Other Settings: Make sure to check “Enable access-based enumeration”.



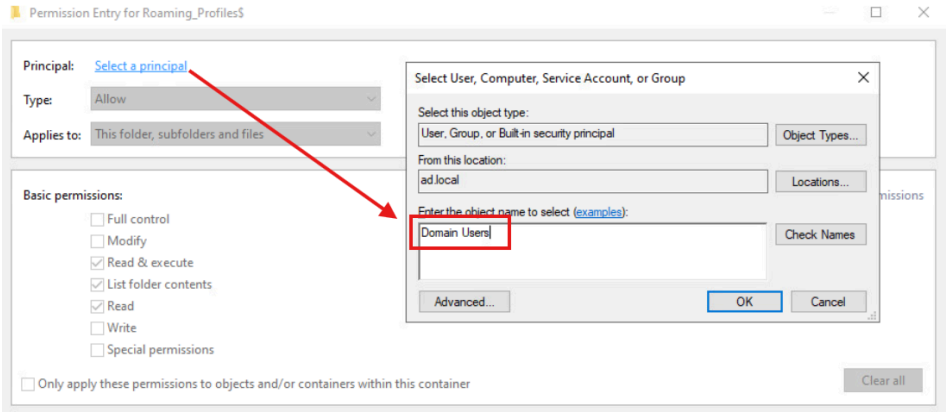
Permissions: Click “Customize Permissions”.



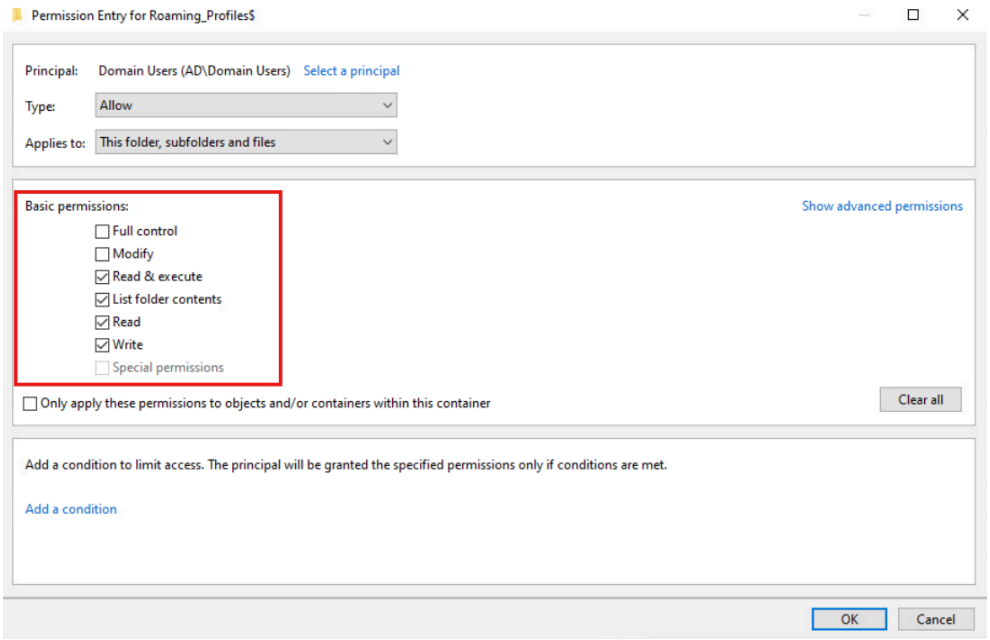
“Add” a new permission.



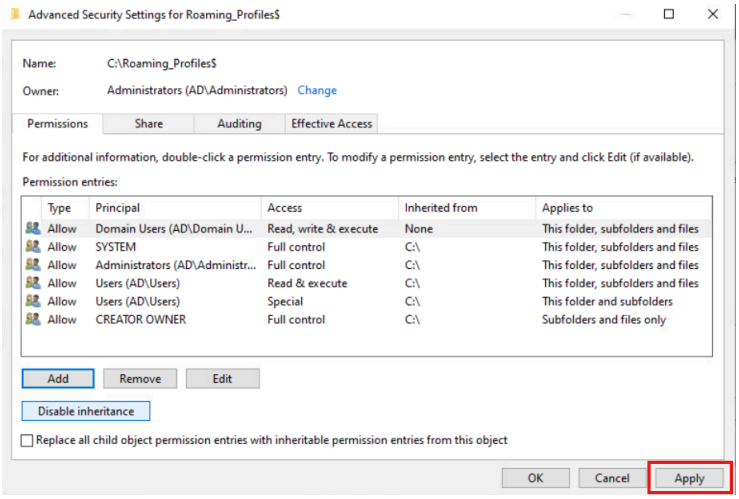
Click “Select a principal” and then enter “Domain Users” in the pop-up window. Select “Check Names” to validate your selection and then choose “Ok” to lock in configurations.



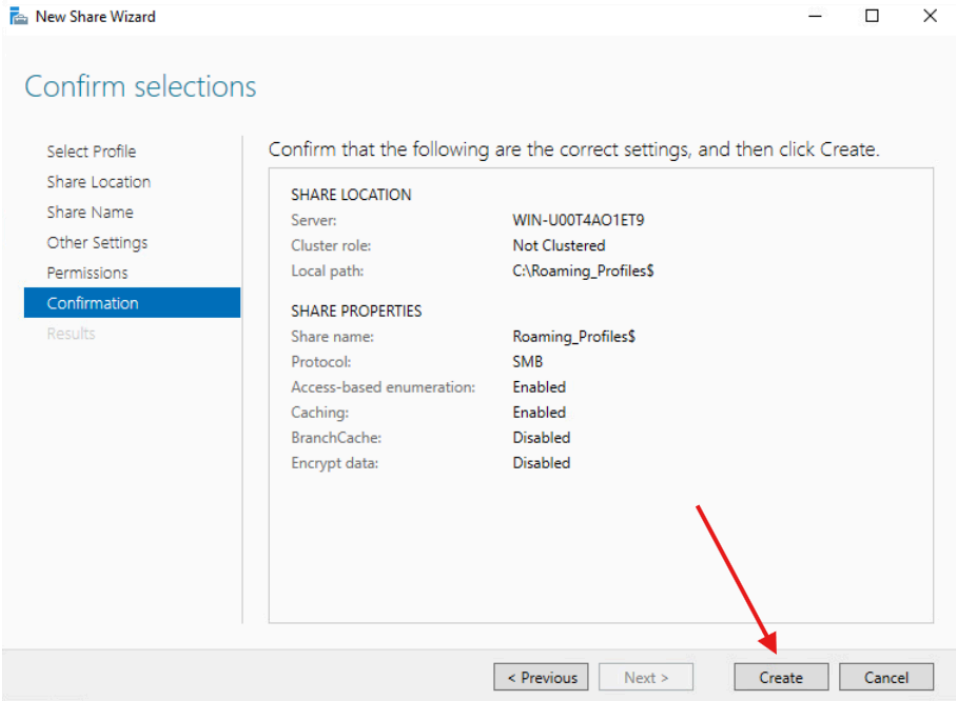
Ensure the following permissions are set so AD users have close to full access to their own file systems. Click “Ok” when done!



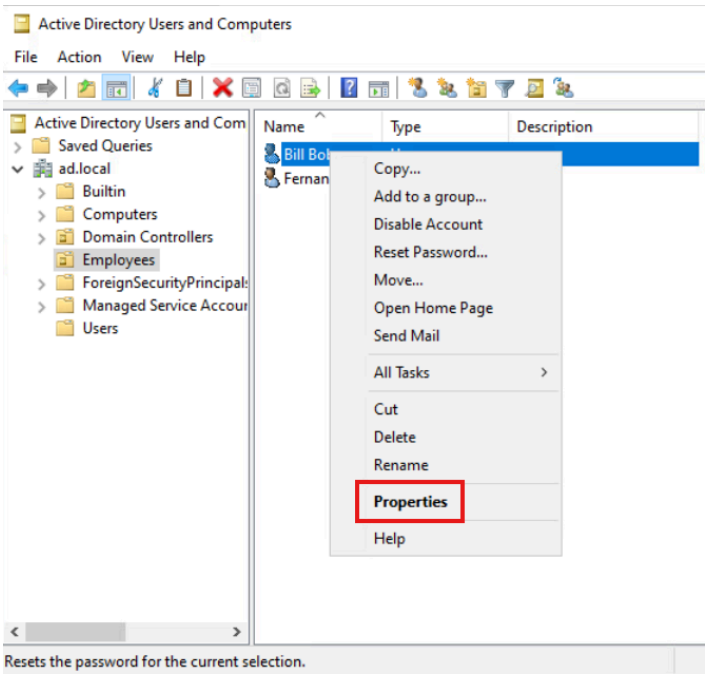
In the original popup-window select “Apply” and then “Ok”.

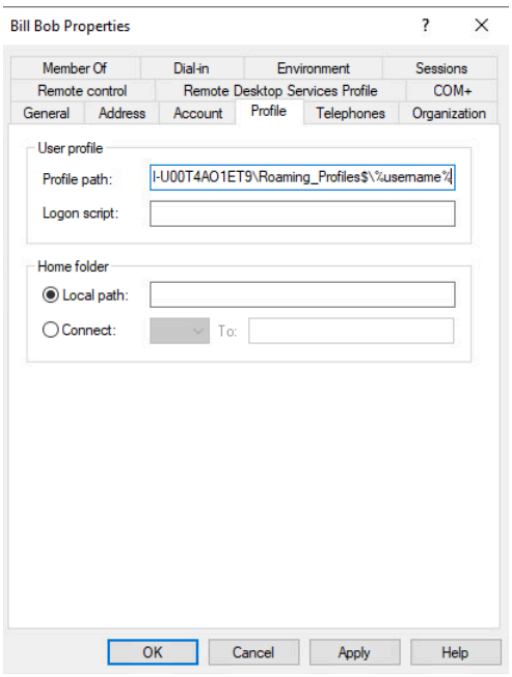


Confirmation: Click "Create" when ready.



Navigate to **Tools > Active Directory Users and Computers > Employees**. Then, right click on a user and select "Properties". Within the popup window, open the "Profile" section and specify the following Profile path: \\server_name\profile_folder_name\%username%.
I used: **\\WIN-U00T4AO1ET9\Roaming_Profiles$\%username%**





Click "Apply" and "Ok" when done!

Project 8: VPN Server

Below contains steps outlining the configuration of a **WireGuard** service on my **pfSense firewall** (**10.161.21.4**). The VPN server will be used to allow approved external clients to connect to an **internal VPN subnet** enabling them to access select internal services. In a real world scenario, this is very helpful for remote work since employees can use a VPN to securely connect to the organization's network from anywhere.

VPN Information
pfSense Firewall: **10.161.21.4**
WireGuard Port: 51820 (default)
Internal VPN Subnet: 192.168.3.0/24
WAN Client IP: **10.161.21.20**
Internal Client IP Assignment: **192.168.3.2**

pfSense Configuration

I will be using WireGuard to host a vpn service on my pfSense firewall.

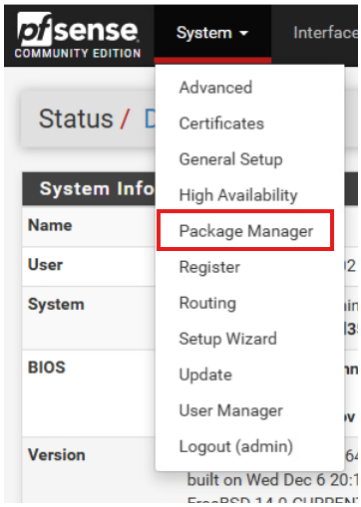
Install WireGuard

WireGuard is a modern, lightweight VPN protocol that operates in the Linux kernel for high performance and minimal overhead, making it ideal for both mobile and server environments. As a VPN server, WireGuard can be deployed to allow remote clients to securely access private networks, route internet traffic through encrypted tunnels, or connect multiple networks across locations.

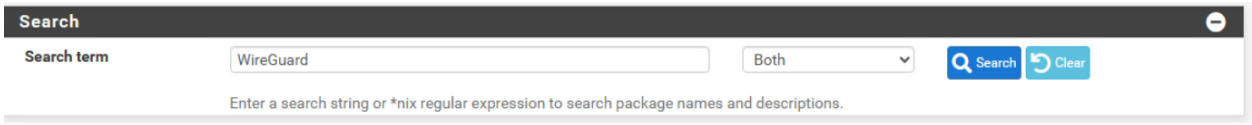
I will create a 3rd subnet managed by pfSense to facilitate my VPN server.
LAN: 192.168.1.0/24
DMZ: 192.168.2.0/24
VPN: 192.168.3.0/24

Login to the pfSense admin dashboard within a browser at http://192.168.1.1

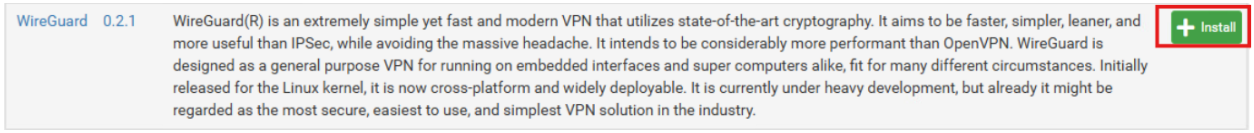
Navigate to **System > Package Manager > Available Packages**



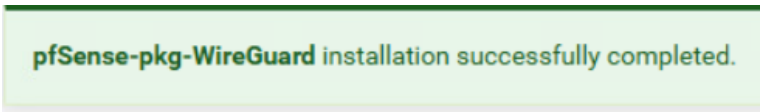
Search for "WireGuard"



Once the WireGuard package is located, click **Install**.

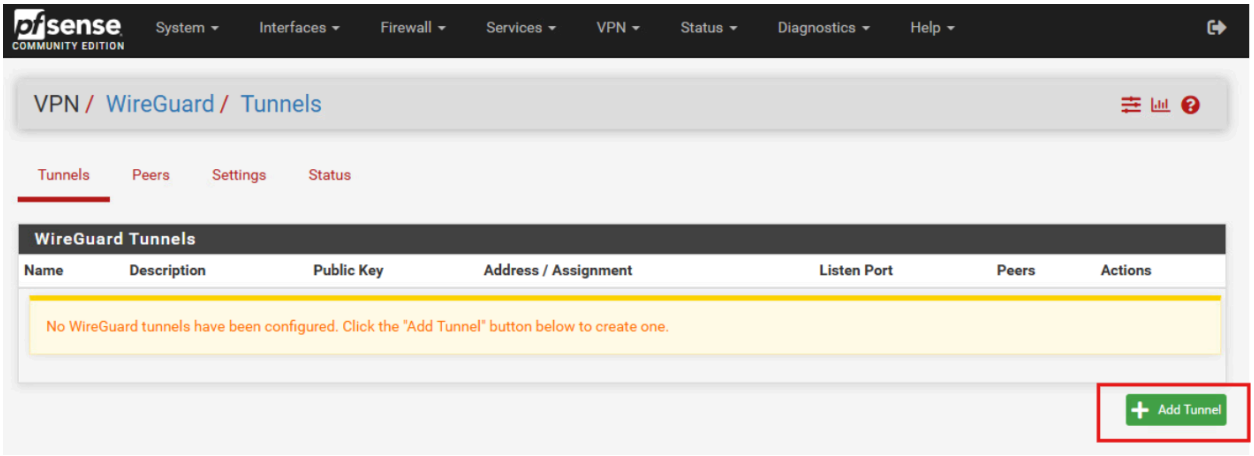


Confirm the installation.



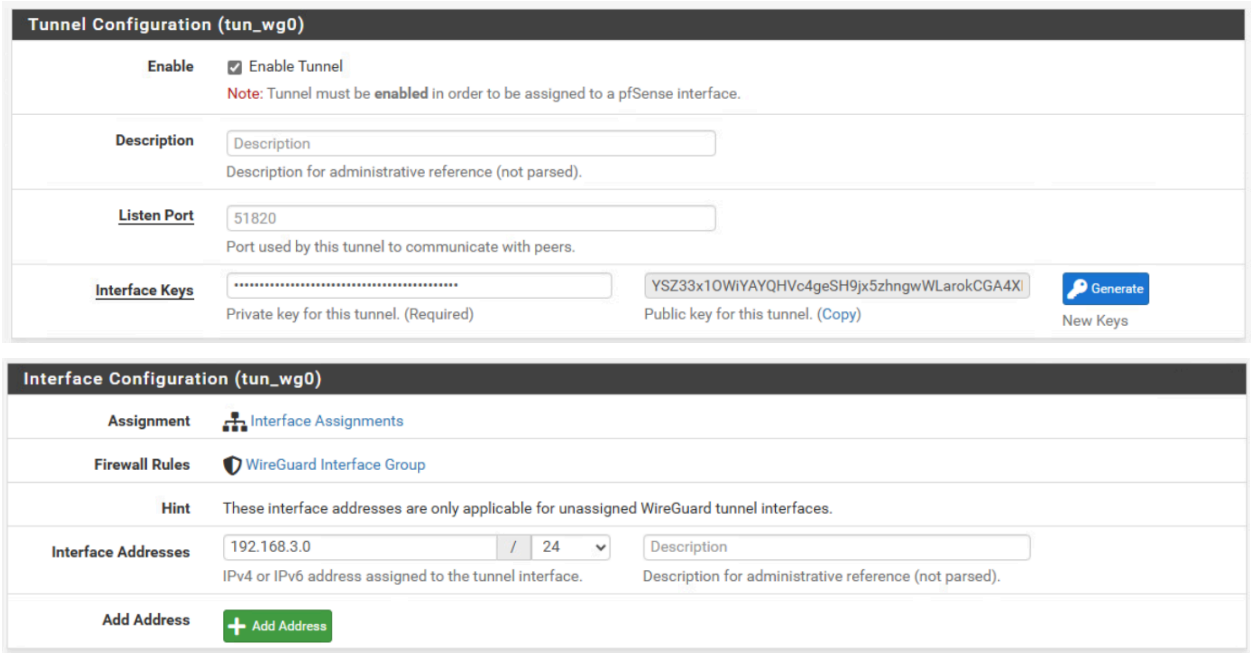
Create WireGuard Tunnel

Navigate to **VPN > WireGuard** using the top bar and click **Add Tunnel**.

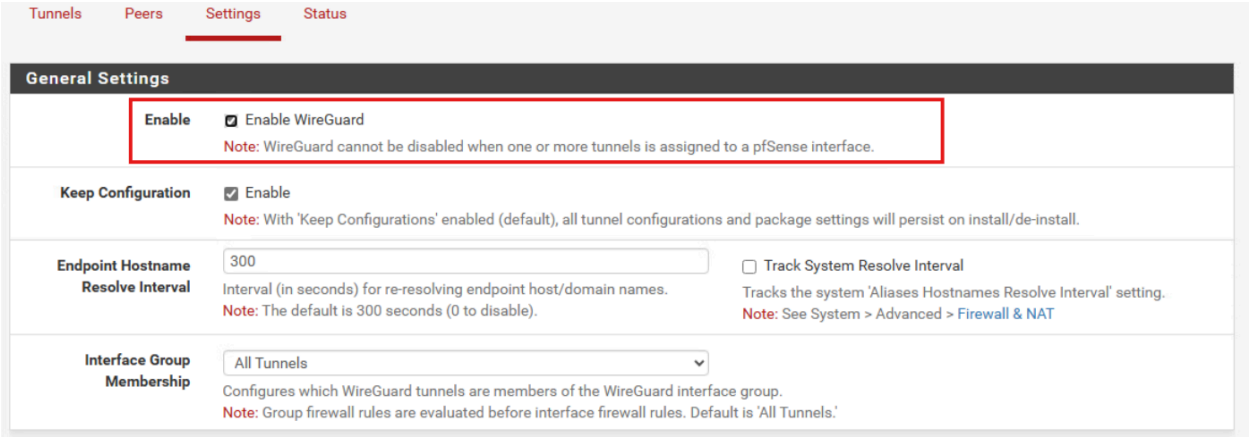


WireGuard Configuration

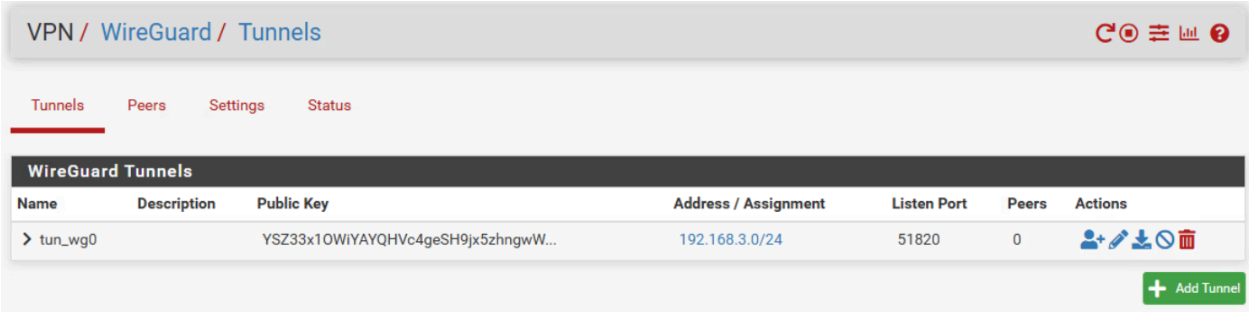
- **Port:** 51820
- **Public Key:** YSZ33x1OWiYAYQHVc4geSH9jx5zhngwWLarokCGA4XI=



Navigate to the **Settings** tab and “Enable WireGuard”.



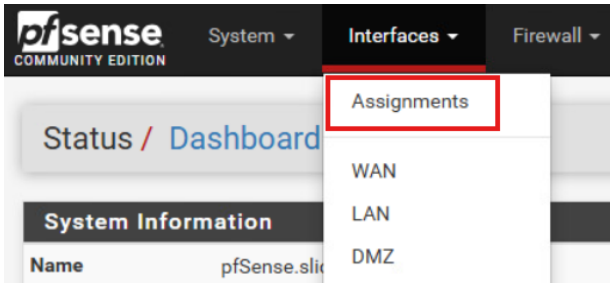
Result:



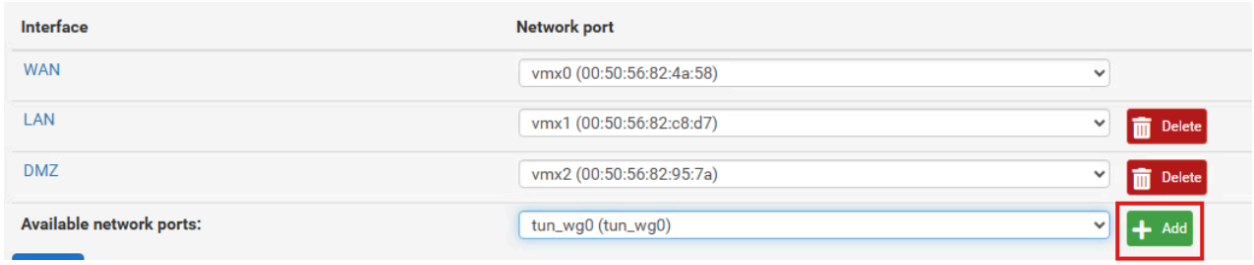
Interface Assignment

I now need to configure the network interface for WireGuard.

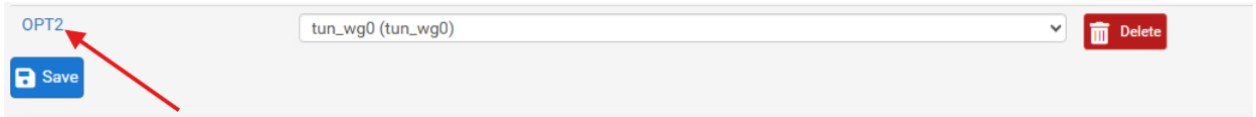
Navigate to **Interfaces > Assignments** using the top bar



Select **Add** to enable the VPN interface.

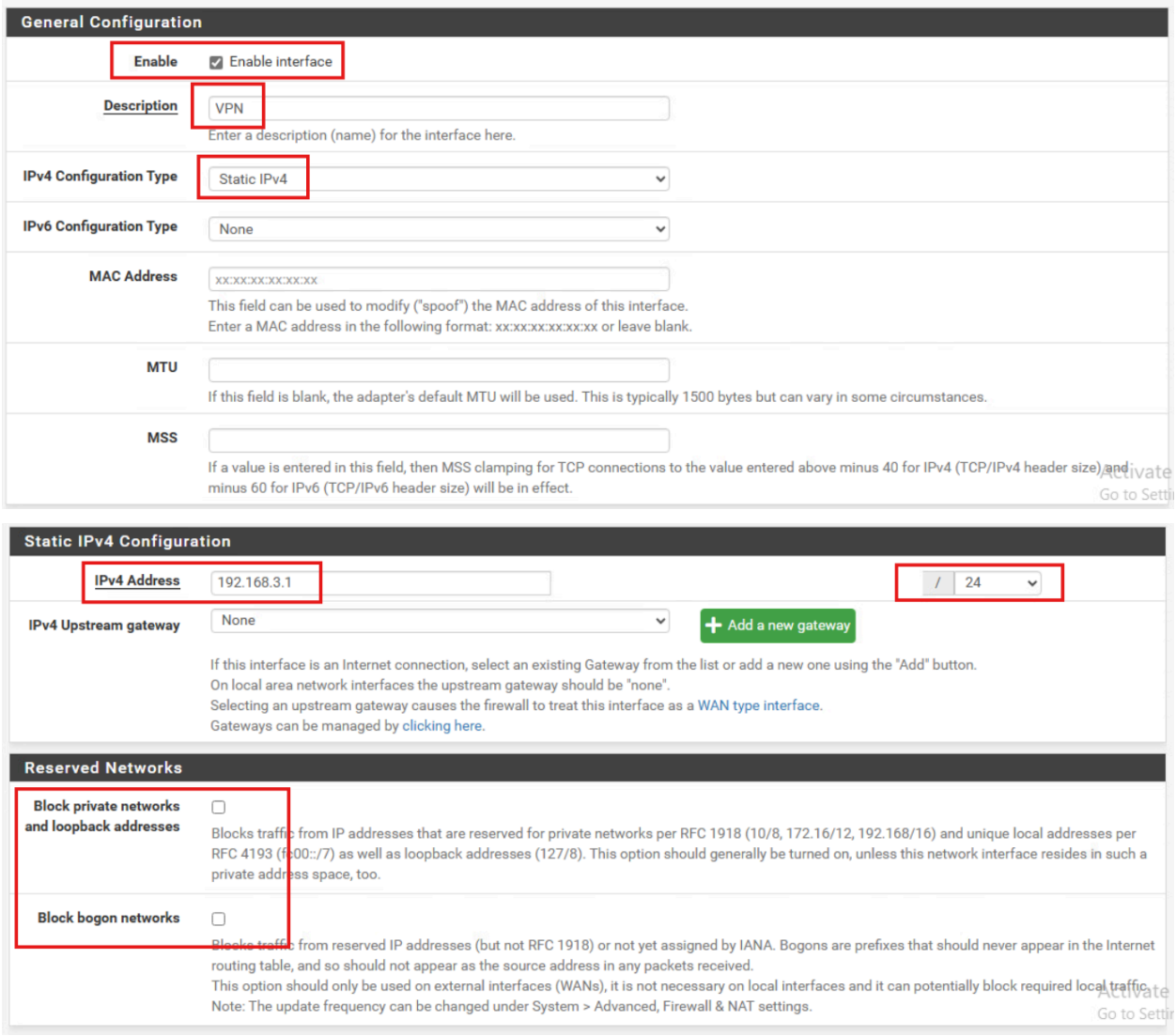


Click on the new interface name “OPT2” to open the configuration menu.



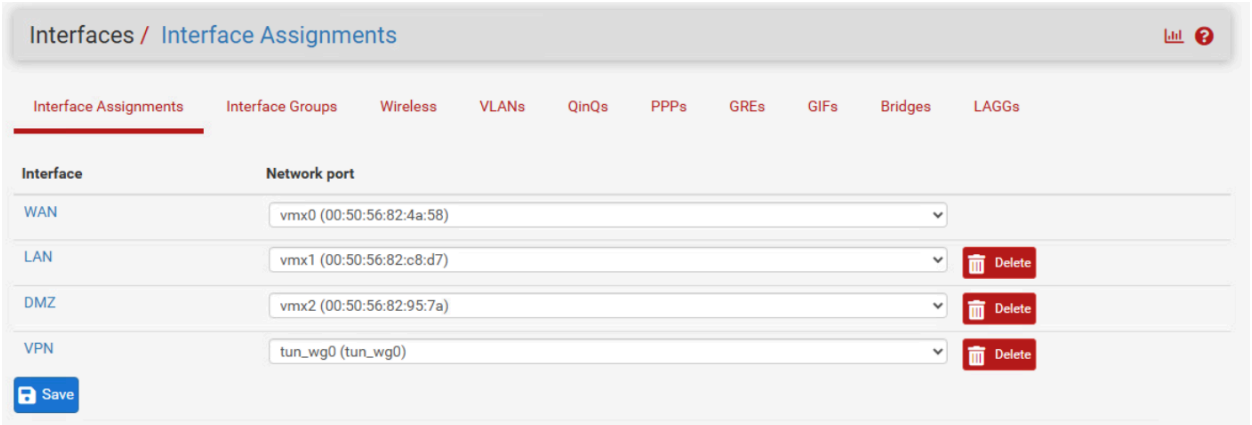
VPN Interface Configuration

- Enable the Interface
- Static IPv4
- IP Range: **192.168.3.1/24**
- Keep “Block private networks” and “Block bogon networks” unchecked



Click **Save** at the bottom when you are done and then **Apply Changes**.

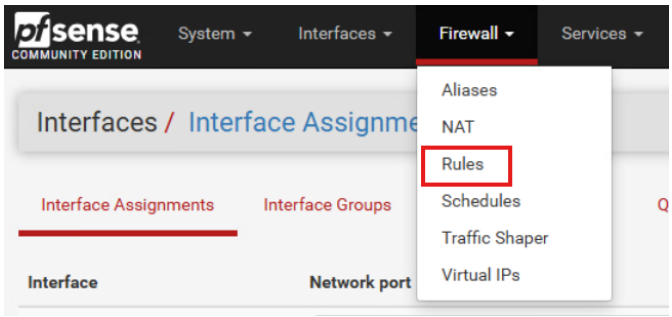
Result:



VPN Rules

I want the VPN subnet to not allow any communication to the LAN and allow all communication to the DMZ.

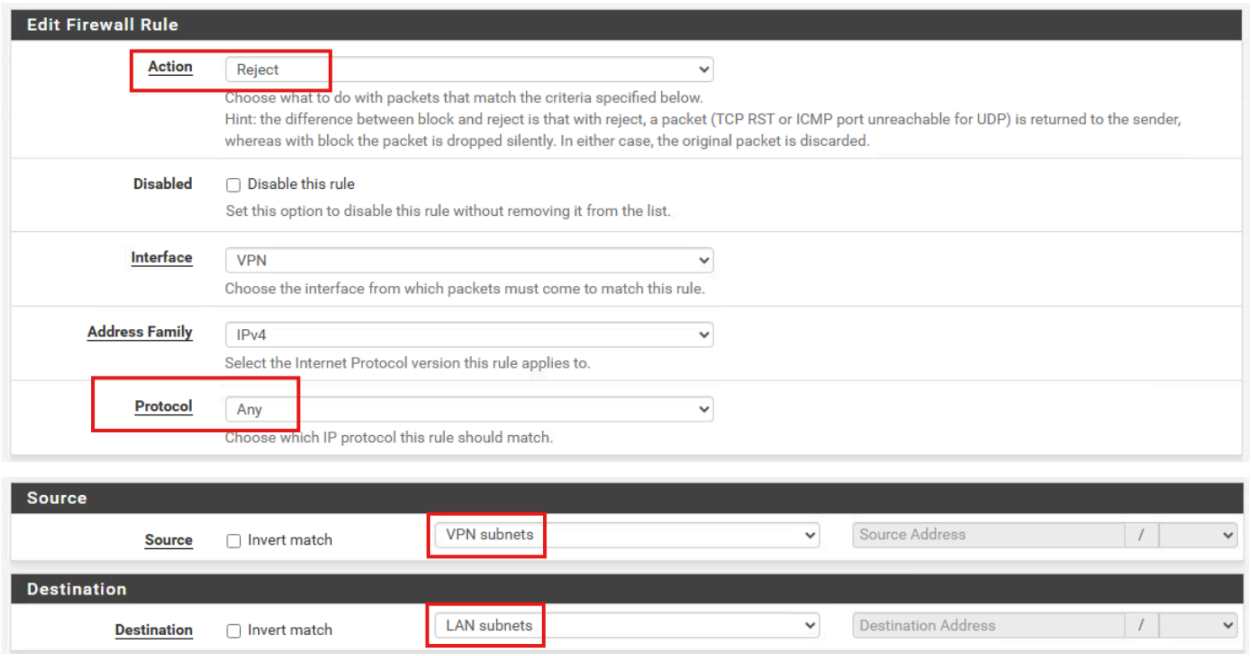
Navigate to **Firewall > Rules** using the top bar.



Within the **VPN** tab, create new firewall rules.

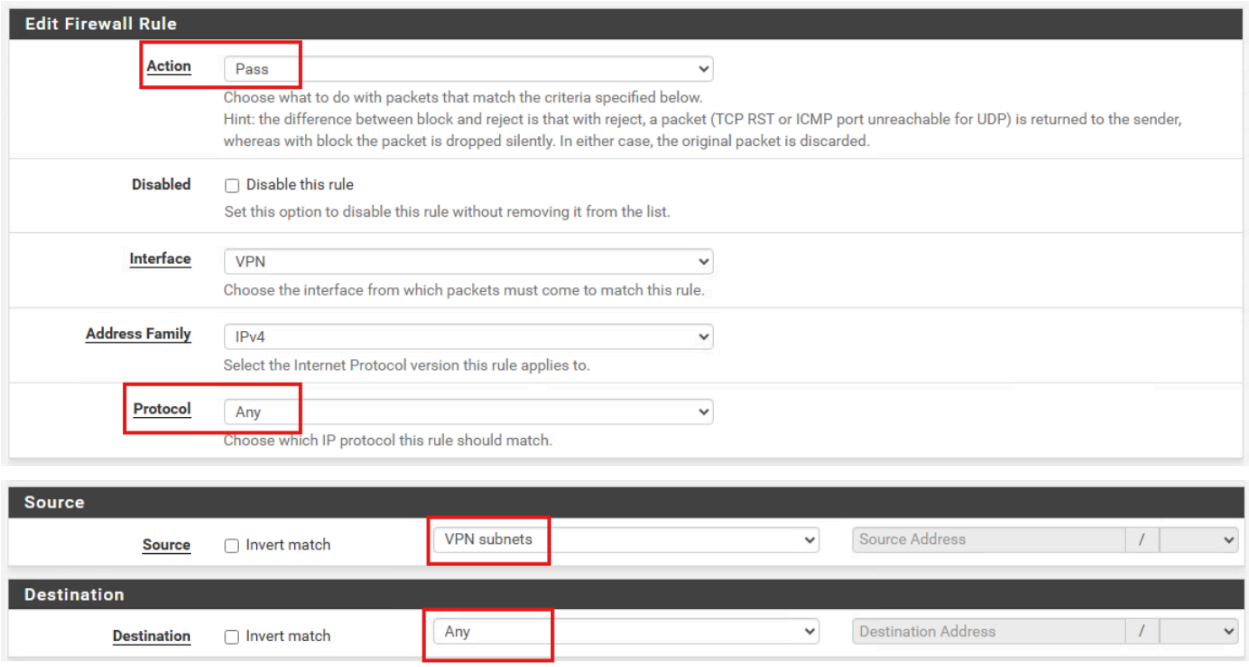
Rule 1: REJECT ALL to LAN

- Source: VPN Subnets
- Destination: LAN Subnets
- This rule will block all traffic traveling to the LAN



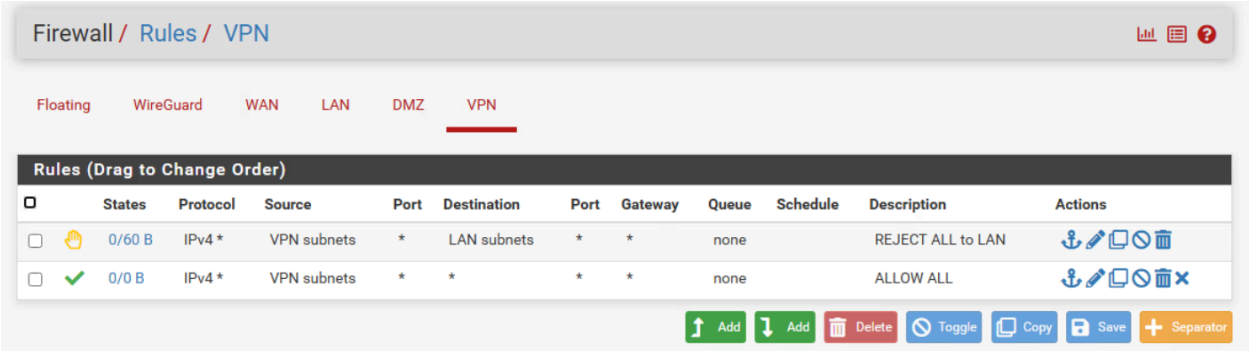
Rule 2: ALLOW ALL

- Source: VPN Subnets
- Destination: Any (*)
- This rule will allow the VPN devices to communicate with the WAN and DMZ



Final VPN Firewall Rules

- NOTE: The REJECT ALL to LAN rule is listed as first priority. This will ensure that traffic to the LAN will not be allowed!

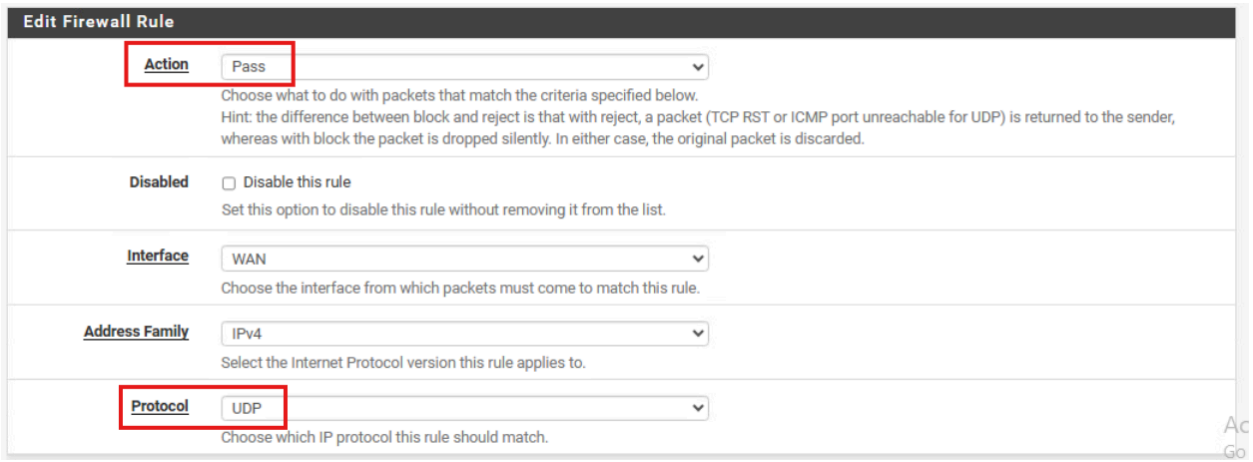


WAN Rules

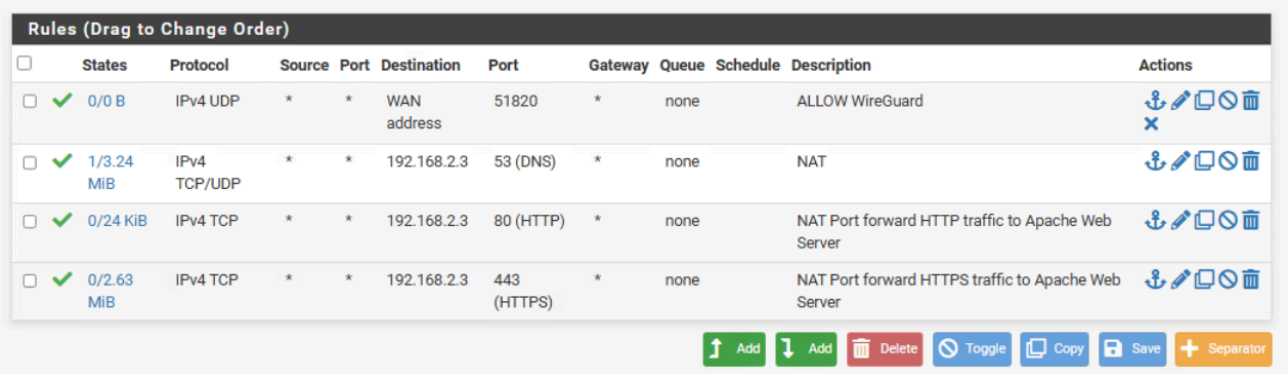
I need to allow VPN traffic from WAN clients into pfSense so they can properly connect.

Rule: ALLOW WireGuard VPN

- Pass
- Protocol: UDP
- Source: Any (*)
- Destination: WAN Address (pfSense public IP: 10.161.21.4)
- Port: 51820 (WireGuard Port)



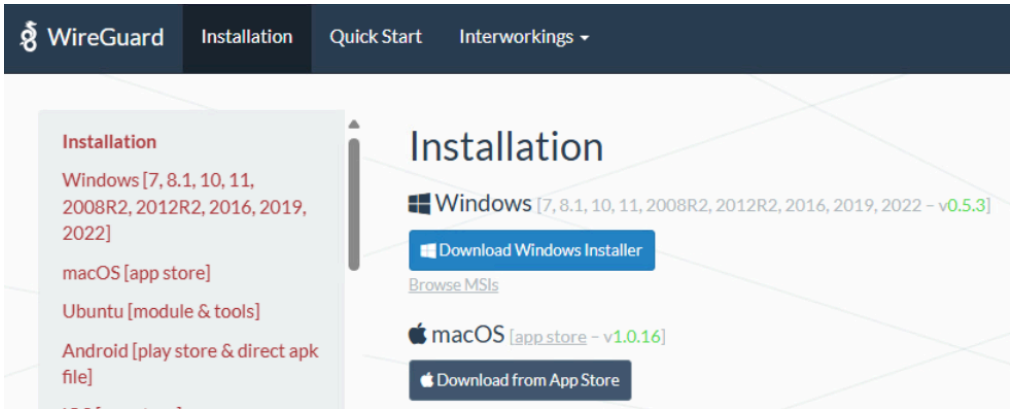
New WAN Firewall Rules



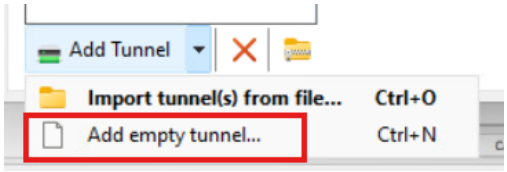
WireGuard Client Setup

In order to connect to the newly created WireGuard VPN service on pfSense, we have to install the proper client software on our WAN device. There is no need to install the software on the LAN workstations since they are already on the internal network.

Login to your WAN client device
Open a browser and navigate to https://www.wireguard.com/install/
Download the Windows Installer



Open the WireGuard application and go to the bottom left corner. Open the dropdown menu and select "Add empty tunnel…"

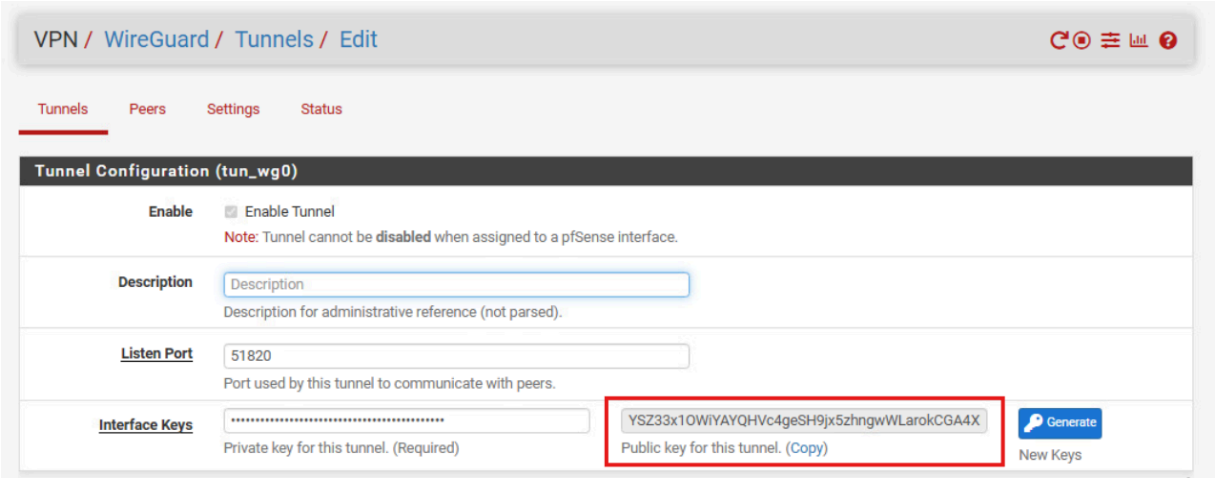


Public Keys

Client Public Key
Add a name for your tunnel and remember your client's public key.
Tunnel Name: **pfSense**
Public Key: AqEDALWCBVqLzbh8YEI9PRokwWBzH++0J7p0EGaJ/g4=

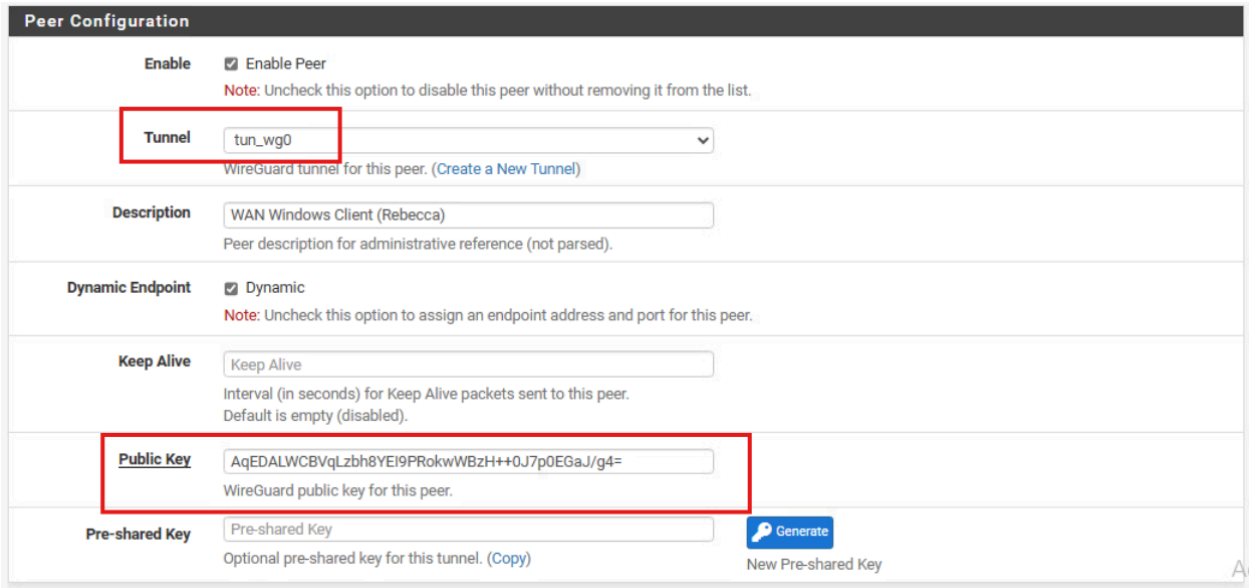
Server Public Key

Go back to pfSense and access your WireGuard VPN Tunnel to get the server's public key.
WireGuard Server Public Key: YSZ33x1OWiYAYQHVc4geSH9jx5zhngwWLarokCGA4XI=
pfSense public WAN IP and Port: **10.161.21.4:51820**



Peer Creation

Stay in pfSense and create a peer connection by navigating to **VPN > WireGuard > Peers** and selecting **Add Peer**.
Note: Public Key is the Windows Client public key!



Client Tunnel Configuration

Navigate back to your client device and finish configuring the new tunnel.

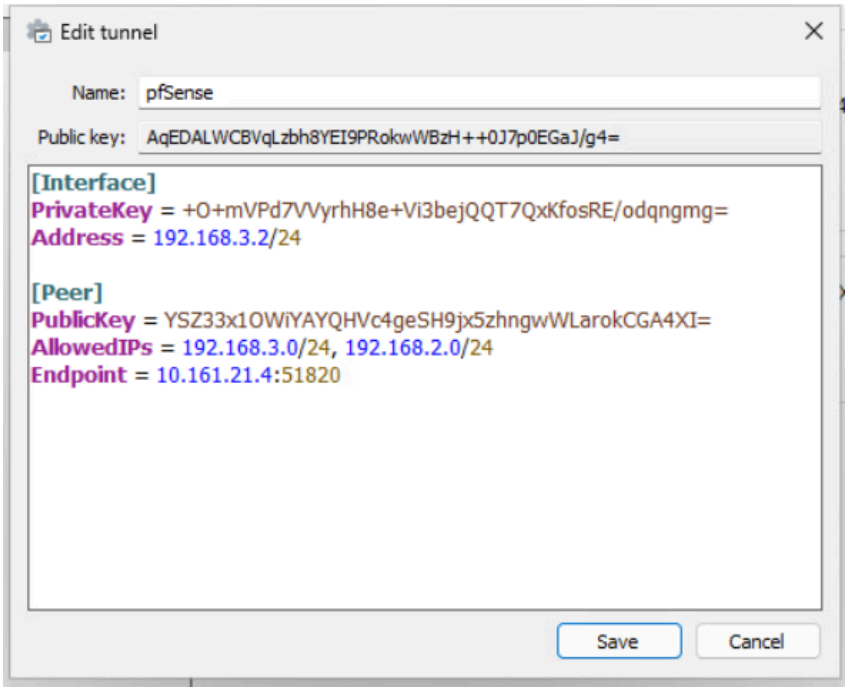
- [INTERFACE]
  - Public Key = the static internal IP assignment when the client is connected
- [PEER]
  - Public Key = pfsense's public key for WireGuard
  - Allowed IPs = the allowed subnets for traffic to route through
  - Endpoint = public socket to connect to the VPN -> pfSenseIP:WireGuard Port

Final Tunnel File

[Interface]
PrivateKey = +O+mVPd7VVyrhH8e+Vi3bejQQT7QxKfosRE/odqngmg=
Address = 192.168.3.2/24

[Peer]
PublicKey = YSZ33x1OWiYAYQHVc4geSH9jx5zhngwWLarokCGA4XI=
AllowedIPs = 192.168.3.0/24, 192.168.2.0/24
Endpoint = 10.161.21.4:51820

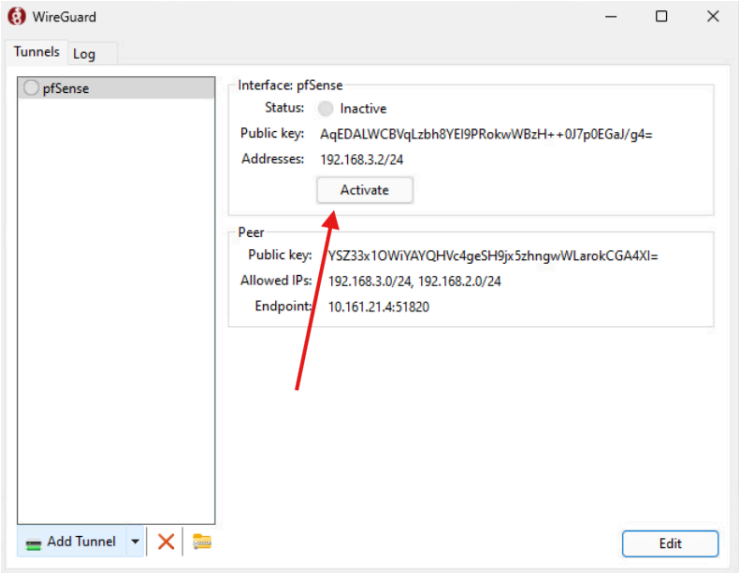
Note: The AllowedIPs section specifies which subnets will be routed through the VPN tunnel. In this configuration, the client device will route traffic destined for 192.168.3.0/24 and 192.168.2.0/24 through the WireGuard tunnel. All other traffic will be sent via the device's regular network interface (NIC), effectively creating a "split tunnel" setup.



VPN Testing

Client Testing

Once you have a proper tunnel file on the client device and a configured peer on pfSense, **Activate** the tunnel within the WireGuard application.



Once the tunnel is active, open a terminal to verify you are within the private VPN subnet with the command: **ipconfig**

You should be able to see 2 network interfaces with an internal VPN IP and a personal IP.

```
PS C:\Users\Rebecca> ipconfig

Windows IP Configuration


Unknown adapter pfSense:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.3.2
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . : slick.firewall
   Link-local IPv6 Address . . . . . : fe80::d6d:cc37:5d7b:1790%4
   IPv4 Address. . . . . . . . . . . : 10.161.21.20
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::250:56ff:fe82:c8d7%4
                                       10.161.21.1

PS C:\Users\Rebecca>
```

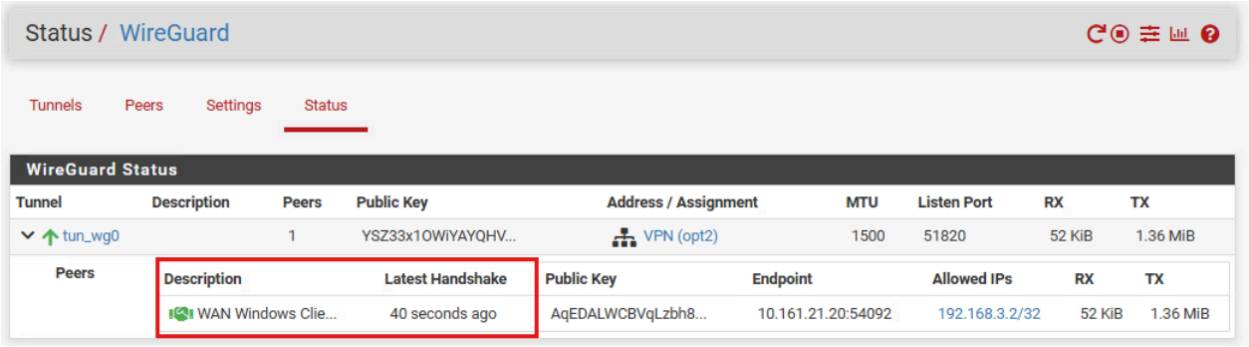
After verifying you are on the internal VPN subnet, try pinging a service in the DMZ such as the web server (**192.168.2.2**) and then try pinging the LAN gateway (**192.168.1.1**).

The DMZ ping should succeed and the LAN ping should fail.

```
PS C:\Users\Rebecca> ping 192.168.2.1

Pinging 192.168.2.1 with 32 bytes of data:
Reply from 192.168.2.1: bytes=32 time=6ms TTL=64
Reply from 192.168.2.1: bytes=32 time<1ms TTL=64
Reply from 192.168.2.1: bytes=32 time=1ms TTL=64
Reply from 192.168.2.1: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.2.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 6ms, Average = 1ms
PS C:\Users\Rebecca> ping 192.168.1.1

Pinging 192.168.1.1 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.1.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Server Testing

Within pfSense, navigate to **VPN > WireGuard > Status** and select the dropdown menu for your tunnel. This will display current connections and successful VPN handshakes.



Appendix

The 3 Most Important Skills I have learned from CS 4400

1. Firewall and Local Network Configuration

Early in the course, I was tasked with configuring a pfSense firewall. Although I have participated in numerous Cyber Defense competitions through the UNISEC Cyber Security club at UNI, I had never been the teammate responsible for setting up a network-level firewall. My experience was limited to host-based firewalls like UFW for Linux or Windows Defender. Booting from the pfSense ISO, setting it up via the terminal, and then transitioning to the browser GUI was a significant learning experience. It provided a refresher on networking concepts such as subnetting, as I had to create two main internal subnets: LAN and DMZ. pfSense also introduced me to creating firewall rules, NAT port forwarding from the WAN into the internal subnets, and setting up services hosted on the firewall, such as the DNS Resolver, DHCP for the LAN, and a WireGuard VPN service. It was incredibly rewarding to finally put computer networking concepts into practice, physically seeing machines and services populate internal networks and connect with each other. Overall, having experience in creating my own Home Lab/Network Cluster will be a great topic to discuss during future job interviews!

2. DNS Configuration

After completing the firewall setup, properly configuring DNS was crucial for ensuring a fully connected network cluster. I had never touched DNS settings on personal devices nor had I ever configured a standalone DNS server before taking this course. My experience was limited to understanding how DNS queries are forwarded through nameservers worldwide to resolve domain names. The progression from setting up an internal caching DNS service on pfSense to installing and configuring a standalone DNS server using BIND was highly beneficial. I started with configuring a simple DNS caching mechanism and then advanced to setting up FQDN mappings using Forward and Reverse lookup zones. DNS setup also involved getting familiar with the nslookup and dig commands. nslookup is used to query DNS servers to obtain domain name or IP address mapping information, while dig provides detailed information about DNS queries and responses, allowing for more advanced troubleshooting and analysis.

3. Active Directory Configuration

The most tedious task of this course was setting up Active Directory on the Windows server. I learned that setting up services on Windows devices involves a lot of “wizards” and GUIs while services on Linux involve the command line and simple configuration files. I prefer the former but it was definitely beneficial to set up AD since it is a common enterprise deployment. For example, my own university uses AD for everything! After completing the initial installation and designating the Windows Server as a Domain Controller, it was fascinating to set up Forward and Reverse lookup zones. Active Directory actually uses DNS to connect to and authenticate client workstations. The setup concluded with creating a new forest and establishing an Organizational Unit (OU). Having some experience with AD will help with enterprise scenarios.