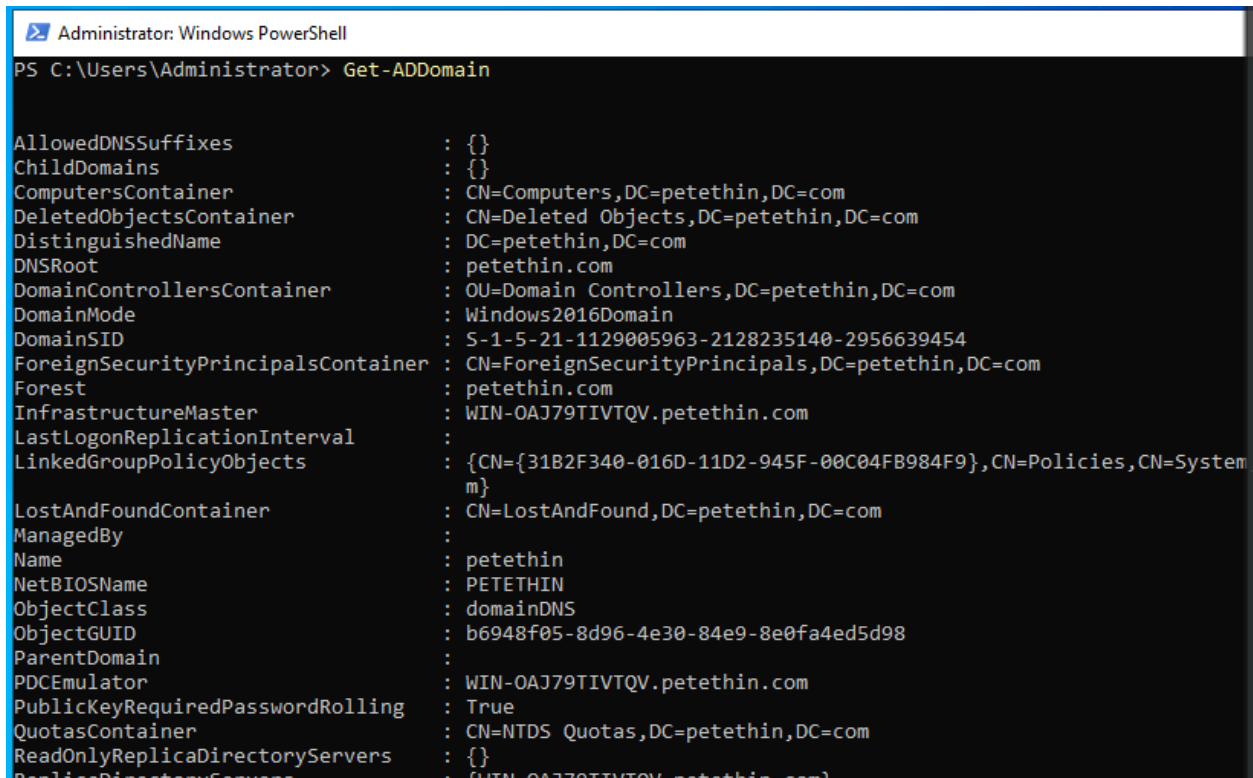


I haven't documented my homelab ever before so I thought I would start.

Current services running in my homelab: SSH, SFTP

6/23:

I installed windows server 2022 desktop onto a virtual machine in virtualbox. The next thing I did was install Active Directory Domain Services. I want to make this a simulated small business network so I thought that configuring a domain controller would be good. This is something I have done in the past but not really anything more than one vm connected to a domain controller so I wanted to improve my overall homelab. Below I will post a screenshot showing my newly created domain.

A screenshot of a Windows PowerShell window titled "Administrator: Windows PowerShell". The command prompt shows the command "Get-ADDomain" being executed. The output is a list of domain-related properties and their values for a domain named "petethin.com".

```
PS C:\Users\Administrator> Get-ADDomain

AllowedDNSSuffixes      : {}
ChildDomains            : {}
ComputersContainer      : CN=Computers,DC=petethin,DC=com
DeletedObjectsContainer : CN=Deleted Objects,DC=petethin,DC=com
DistinguishedName       : DC=petethin,DC=com
DNSRoot                 : petethin.com
DomainControllersContainer : OU=Domain Controllers,DC=petethin,DC=com
DomainMode              : Windows2016Domain
DomainSID               : S-1-5-21-1129005963-2128235140-2956639454
ForeignSecurityPrincipalsContainer : CN=ForeignSecurityPrincipals,DC=petethin,DC=com
Forest                  : petethin.com
InfrastructureMaster     : WIN-0AJ79TIVTQV.petethin.com
LastLogonReplicationInterval : 
LinkedGroupPolicyObjects : {CN={31B2F340-016D-11D2-945F-00C04FB984F9},CN=Policies,CN=System
m}
LostAndFoundContainer    : CN=LostAndFound,DC=petethin,DC=com
ManagedBy               : 
Name                     : petethin
NetBIOSName              : PETETHIN
ObjectClass              : domainDNS
ObjectGUID               : b6948f05-8d96-4e30-84e9-8e0fa4ed5d98
ParentDomain             : 
PDCEmulator              : WIN-0AJ79TIVTQV.petethin.com
PublicKeyRequiredPasswordRolling : True
QuotasContainer          : CN=NTDS Quotas,DC=petethin,DC=com
ReadOnlyReplicaDirectoryServers : {}
ReplicaDirectoryServers  : {WIN-0AJ79TIVTQV.petethin.com}
```

My next goal is to ensure connectivity between my raspberry pi server, and my newly created windows domain controller.

I made sure to configure a static IP address because using a static IP address on a server running business critical services is best practice(This is a homelab however I still want to use best practices) After configuring the static IP address, I tested connectivity by pinging the windows server from my rpi that I ssh'd into and then I pinged the raspberry pi from my domain controller server. They both sent pings with no packet loss.

6/24:

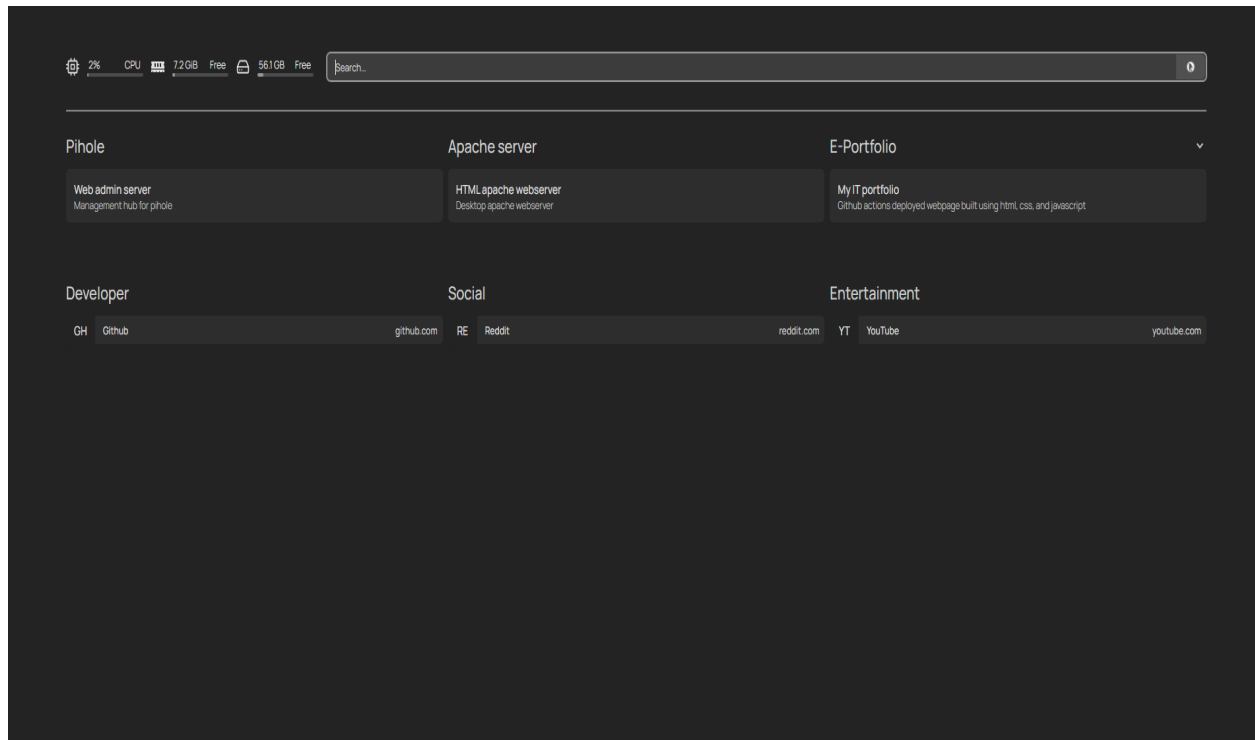
I made sure to create a snapshot of the domain controller virtualized server and the client that will be connected to the domain. I'm planning on putting the snapshots onto an external hard drive, and flash drive to have multiple backups just in case. I want to implement my raspberry pi into the windows side of my homelab. I'm going to. I haven't implemented that since I formatted

my micro sd card and flashed freebsd to it. I however didn't like freebsd so I went with raspberry pi os lite. I will soon be reimplementing the pi-hole service and then putting it on the windows domain controller and client to block ads and domains. I will accomplish this by individually setting each device's DNS server as the static IP address of the raspberry pi. I know that I could make every device in my network use this service through my router, however I prefer giving others in my home the choice of whether they would want to use this service or not.

6/28: I setup pi-hole and a webserver to manage my pihole adlists. This took around an hour or so because I already had a static IP address setup on my pi as I primarily use it as a server. My pi also is very quick due to the fact that I use raspberry pi os lite which uses very little resources. I ran these commands to configure it.

1. `sudo apt install git`
2. `git clone --depth 1 https://github.com/pi-hole/pi-hole.git Pi-hole`
3. `cd "Pi-hole/automated install/"`
4. `sudo bash basic-install.sh`

7/1: I wanted a good management hub for my homelab services. I decided to setup homepage through docker. I used docker run to deploy the homepage onto my network using a custom port(I believe I used 3000). I bookmarked my homepage. It is convenient as I can access all my server logins and school logins in a centralized web page. My first ever big homelab project was configuring my own custom homepage for my web servers and school login pages yet it didn't look nearly as good as the homepage does. For my homepage I have my pihole web server used for administration purposes for that server, I also have my desktop webserver and eportfolio listed.



7/2-7/4:

I have done some upgrades with the homepage dashboard to spruce up my homelab. I edited the services.yml configuration file as well as the settings.yml and bookmarks.yml configuration

I also have made a docker container running a minecraft bedrock server. I wanted to host a service that would have actual use besides just for my own purposes. My girlfriend likes to play minecraft on our xbox so I thought setting up a bedrock edition server would be a cool thing to self host and actually have a user besides myself.

I did this by

1. Mkdir minecraft-bedrock-server
2. cd minecraft-bedrock-server
3. touch compose.yml
4. I span it up using a docker compose file with the configuration shown below

version: '3'

services:

bedrock-server:

image: itzg/minecraft-bedrock-server

container_name: bedrock-server

environment:

EULA: "TRUE"

Ports:

- "19132:19132/udp"

Volumes:

- ./data:/data

5. After this I wanted to ensure that it worked and was actually running by testing it on my xbox.

7/5:

I didn't have time to test it on the fourth of July as I was out of town for the most part. After getting back today I did some configuration changes to the server.properties file in my bedrock server directory.

After making the changes I issued the docker ps -a command to list all running docker containers.

I grabbed the docker container id of the current running minecraft server to restart by issuing this command: docker restart <container id>

After restarting the container I ensured it was running by again issuing the docker ps -a command to find that it was healthy but starting. I gave it a few minutes and reissued the command again to check the status to see that it was up and running as well as healthy.

Me and my girlfriend played split screen to test out the server and to my surprise it actually ran pretty efficiently. There wasn't any lagging or stuttering which I expected.

I plan on implementing some changes to the server due to my girlfriend's request such as starting with coordinates, and adding mods to enrich the gameplay.

I also will be implementing the bedrock docker server documentation to my homepage dashboard for efficiency purposes.