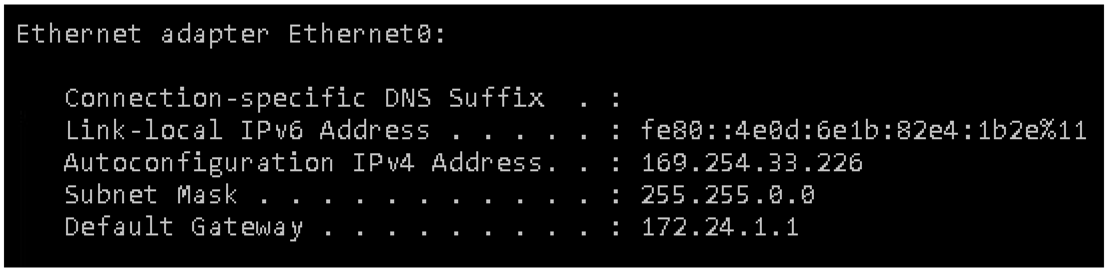
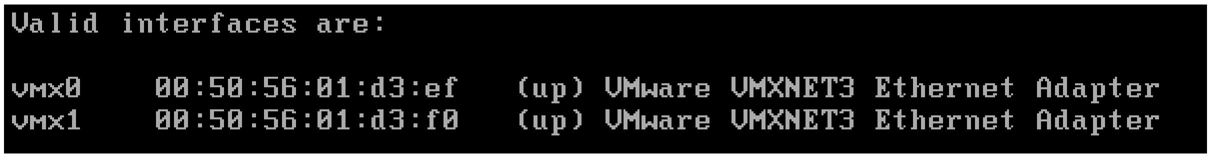
I am having issues connecting the client to the server. I have tried both statically and dynamically allocating the IP address and it keeps returning an Automatic Private IP Addressing (APIPA) address, indicating that DHCP failed to assign a proper IP address.

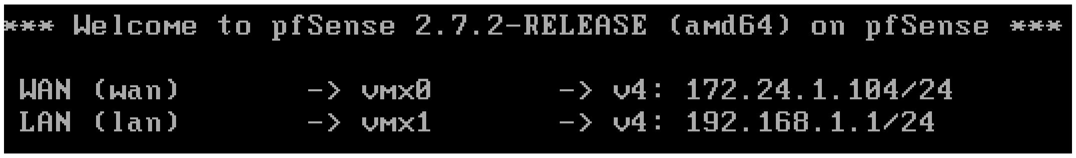


```
Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::4e0d:6e1b:82e4:1b2e%11
   Autoconfiguration IPv4 Address. . : 169.254.33.226
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . : 172.24.1.1
```

Pfsense shows both interfaces as UP and with their correct IP ranges:



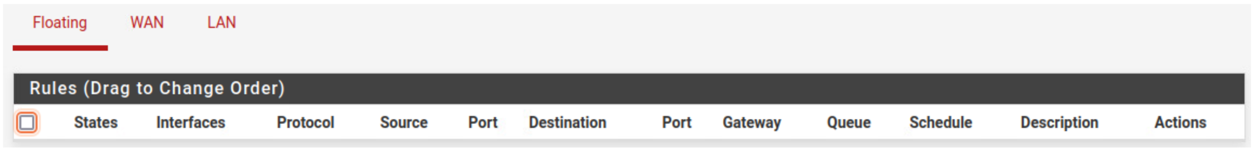
```
Valid interfaces are:

vmx0     00:50:56:01:d3:ef    (up) VMware VMXNET3 Ethernet Adapter
vmx1     00:50:56:01:d3:f0    (up) VMware VMXNET3 Ethernet Adapter
```



```
*** Welcome to pfSense 2.7.2-RELEASE (amd64) on pfSense ***

 WAN (wan)       -> vmx0       -> v4: 172.24.1.104/24
 LAN (lan)       -> vmx1       -> v4: 192.168.1.1/24
```

As a result, any attempts to ping the server return “Destination host unreachable.”

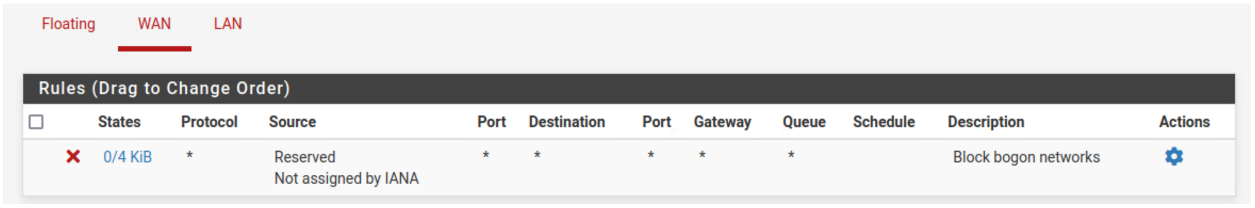
Therefore, while I did get the firewall up and running, I was unable to determine the specific rules required for proper communication. Screenshots of this configuration are below:



Floating   WAN   LAN

Rules (Drag to Change Order)

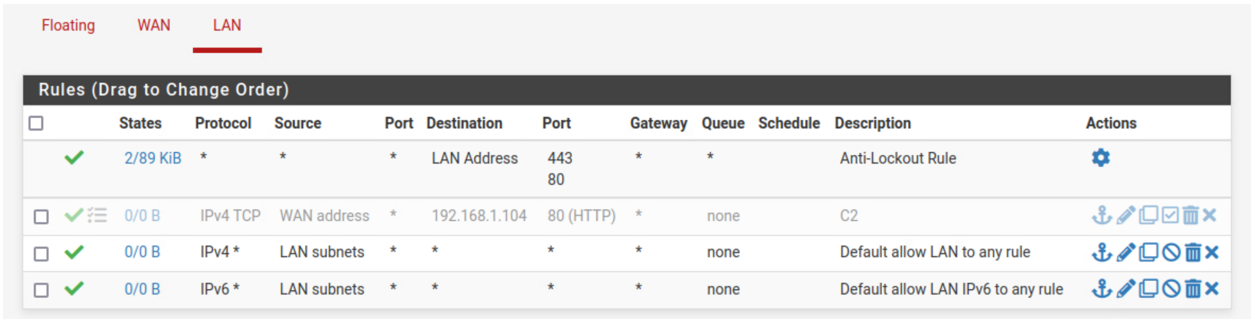
☐	States	Interfaces	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions



Floating   WAN   LAN

Rules (Drag to Change Order)

☐		States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
	✗	0/4 KiB	*	Reserved Not assigned by IANA	*	*	*	*	*		Block bogon networks	⚙



Floating   WAN   LAN

Rules (Drag to Change Order)

☐		States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
	✓	2/89 KiB	*	*	*	LAN Address	443 80	*	*		Anti-Lockout Rule	⚙
☐	✓	0/0 B	IPv4 TCP	WAN address	*	192.168.1.104	80 (HTTP)	*	none		C2	
☐	✓	0/0 B	IPv4 *	LAN subnets	*	*	*	*	none		Default allow LAN to any rule	
☐	✓	0/0 B	IPv6 *	LAN subnets	*	*	*	*	none		Default allow LAN IPv6 to any rule	

Despite these challenges, I still wrote the corresponding python scripts and ensured overall functionality.

The server script manages client requests, covering tasks like shutting down hosts or clients, retrieving MAC addresses, IP addresses, OS details, and handling file uploads. It operates by monitoring a designated IP address and port, accepting incoming connections, and analyzing user input to generate tailored requests for the client. It handles incoming requests using HTTP GET or POST methods, where GET requests establish initial connections, and POST requests deliver requested data. Through techniques like obfuscating data in headers (like ETag) or using dummy URLs, the server creates an appearance of genuine HTTP requests and responses.

The server script includes functions to handle client requests such as shutting down the host or client script, extracting MAC addresses, IP addresses, OS information, and uploading files. It operates by listening on a specified IP address and port, accepting incoming connections, and analyzing user input to generate tailored requests for the client. It also handles incoming HTTP GET or POST requests, where GET requests inform the server that the client is ready to execute commands, and POST requests deliver requested data. Through techniques like obfuscating data in headers (such as ETag) or using dummy URLs, the server creates an appearance of genuine HTTP requests and responses.

When the client initially connects, it sends a GET request to the server to inform that a connection has been made. It then listens for a legitimate HTTP status code, each of which corresponds with specific commands:

- A 204 No Content response signals the client to shut down a target. The client interprets the first letter of the ETAG to determine if the target is a 'host' (first letter 'h') or 'client' (first letter 'c'). Additionally, it retrieves the Cache-Control: max-age value, which specifies a delay for the shutdown. If the value is 0, the client shuts down the target immediately; otherwise, it waits for the specified duration.
- A 304 Not Modified response informs the client to extract data from the host machine. Similar to the 204 request, the client extracts the first letter of the randomly generated ETag and determines whether to extract MAC, IP, or OS information.
- A 301 Moved Permanently response instructs the client to upload a file. By extracting the URL from the location header and removing the dummy URL ('http://tla.com'), the client obtains the file path to the target file. It then reads the file and sends the data back to the server.

The full scripts can be found in the attached folder.