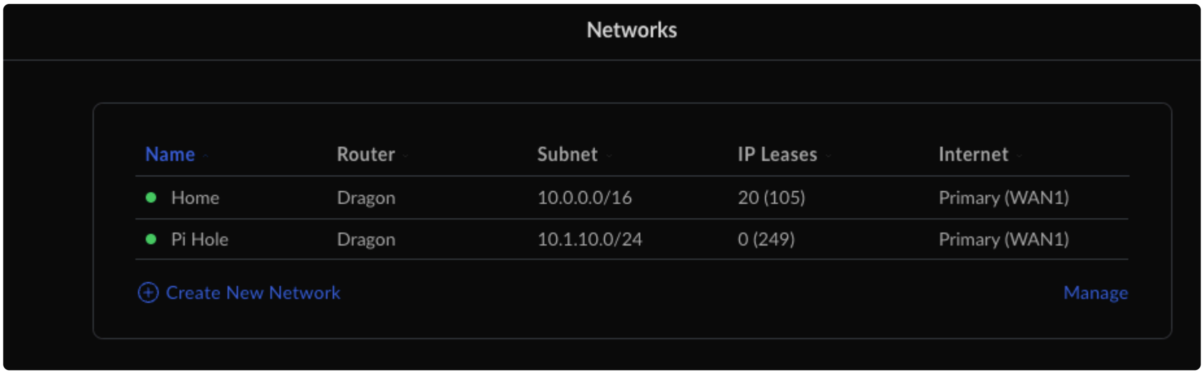
PiHole/Unbound Setup nSpawn

Links

nSpawn YouTube
PiHole and Unbound Setup
PiHole on UDM nSpawn Setup 3.X.X
PiHole on UDM Podman Setup 2.X.X
Unbound on UDM Podman Setup 2.X.X
Setup Video
AD Block Test Site
AD Block Lists Repo

PiHole VLAN Setup

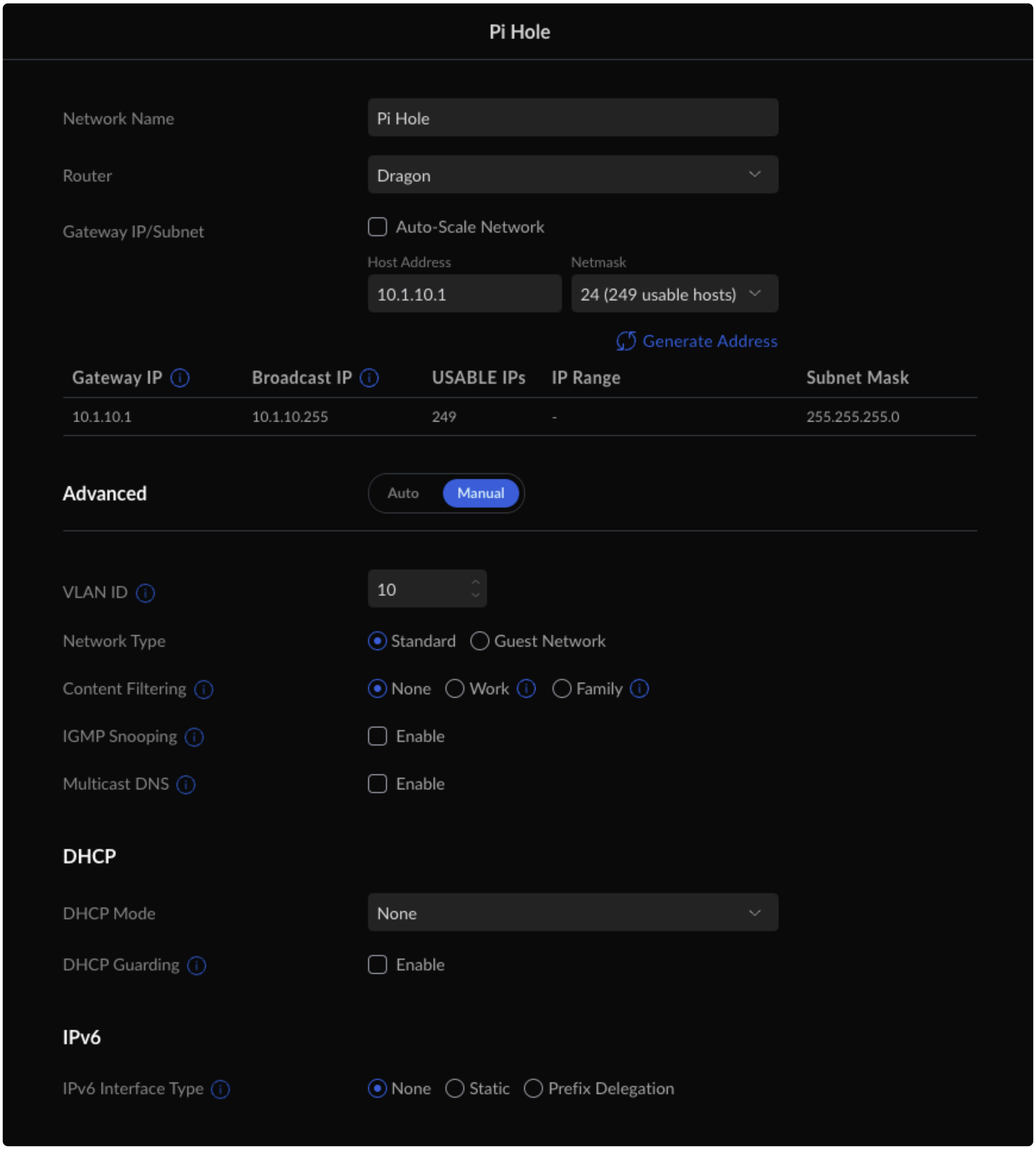
1. Create VLAN network called **PiHole** under **Network** → **Settings** → **Networks**



Networks

Name	Router	Subnet	IP Leases	Internet
● Home	Dragon	10.0.0.0/16	20 (105)	Primary (WAN1)
● Pi Hole	Dragon	10.1.10.0/24	0 (249)	Primary (WAN1)

2. Set the network as per below.



Pi Hole

Network Name: Pi Hole

Router: Dragon

Gateway IP/Subnet: □ Auto-Scale Network

Host Address: 10.1.10.1

Netmask: 24 (249 usable hosts)

Gateway IP	Broadcast IP	USABLE IPs	IP Range	Subnet Mask
10.1.10.1	10.1.10.255	249	-	255.255.255.0

Advanced: Auto [✓ Manual]

VLAN ID: 10

Network Type: ● Standard ○ Guest Network

Content Filtering: ● None ○ Work ○ Family

IGMP Snooping: □ Enable

Multicast DNS: □ Enable

DHCP

DHCP Mode: None

DHCP Guarding: □ Enable

IPv6

IPv6 Interface Type: ● None ○ Static ○ Prefix Delegation

PiHole Setup

1. **machinectl shell debian-custom**

2. **apt -y install curl**

3. **curl -sSL** https://install.pi-hole.net **| PIHOLE_SKIP_OS_CHECK=true bash**

4. Run through PiHole setup.

5. Set Pihole password: **pihole -a -p**

6. PiHole admin page accessible at http://10.1.10.2/admin

7. As a final step, you need to set "**Permit all origins**" in the Pi-Hole Admin to allow requests from more than one hop away (i.e. your LAN clients). Go to **Pi-Hole Admin → Settings → DNS → Permit all origins → Save**.

Unbound Setup

1. **sudo apt install unbound -y**

2. **vi /etc/unbound/unbound.conf.d/pi-hole.conf**

pi-hole.conf

```
server:
# If no logfile is specified, syslog is used
# logfile: "/var/log/unbound/unbound.log"
verbosity: 0

interface: 127.0.0.1
port: 5335
do-ip4: yes
do-udp: yes
do-tcp: yes

# May be set to yes if you have IPv6 connectivity
do-ip6: no

# You want to leave this to no unless you have *native* IPv6. With 6to4 and
# Terredo tunnels your web browser should favor IPv4 for the same reasons
prefer-ip6: no

# Use this only when you downloaded the list of primary root servers!
# If you use the default dns-root-data package, unbound will find it
automatically
#root-hints: "/var/lib/unbound/root.hints"

# Trust glue only if it is within the server's authority
```

```
harden-glue: yes

# Require DNSSEC data for trust-anchored zones, if such data is absent, the
zone becomes BOGUS
harden-dnssec-stripped: yes

# Don't use Capitalization randomization as it known to cause DNSSEC issues
sometimes
# see https://discourse.pi-hole.net/t/unbound-stubby-or-dnscrypt-proxy/9378 for
further details
use-caps-for-id: no

# Reduce EDNS reassembly buffer size.
# IP fragmentation is unreliable on the Internet today, and can cause
# transmission failures when large DNS messages are sent via UDP. Even
# when fragmentation does work, it may not be secure; it is theoretically
# possible to spoof parts of a fragmented DNS message, without easy
# detection at the receiving end. Recently, there was an excellent study
# >>> Defragmenting DNS - Determining the optimal maximum UDP response size for
 DNS <<<
# by Axel Koolhaas, and Tjeerd Slokker (https://indico.dns-oarc.net/event/36/
contributions/776/)
# in collaboration with NLnet Labs explored DNS using real world data from the
# the RIPE Atlas probes and the researchers suggested different values for
# IPv4 and IPv6 and in different scenarios. They advise that servers should
# be configured to limit DNS messages sent over UDP to a size that will not
# trigger fragmentation on typical network links. DNS servers can switch
# from UDP to TCP when a DNS response is too big to fit in this limited
# buffer size. This value has also been suggested in DNS Flag Day 2020.
edns-buffer-size: 1232

# Perform prefetching of close to expired message cache entries
# This only applies to domains that have been frequently queried
prefetch: yes

# One thread should be sufficient, can be increased on beefy machines. In
reality for most users running on small networks or on a single machine, it
should be unnecessary to seek performance enhancement by increasing num-threads
above 1.
num-threads: 1

# Ensure kernel buffer is large enough to not lose messages in traffic spikes
so-rcvbuf: 1m

# Ensure privacy of local IP ranges
private-address: 192.168.0.0/16
private-address: 169.254.0.0/16
private-address: 172.16.0.0/12
private-address: 10.0.0.0/8
private-address: fd00::/8
private-address: fe80::/10
```

3. **service unbound restart**

4. **service unbound status**

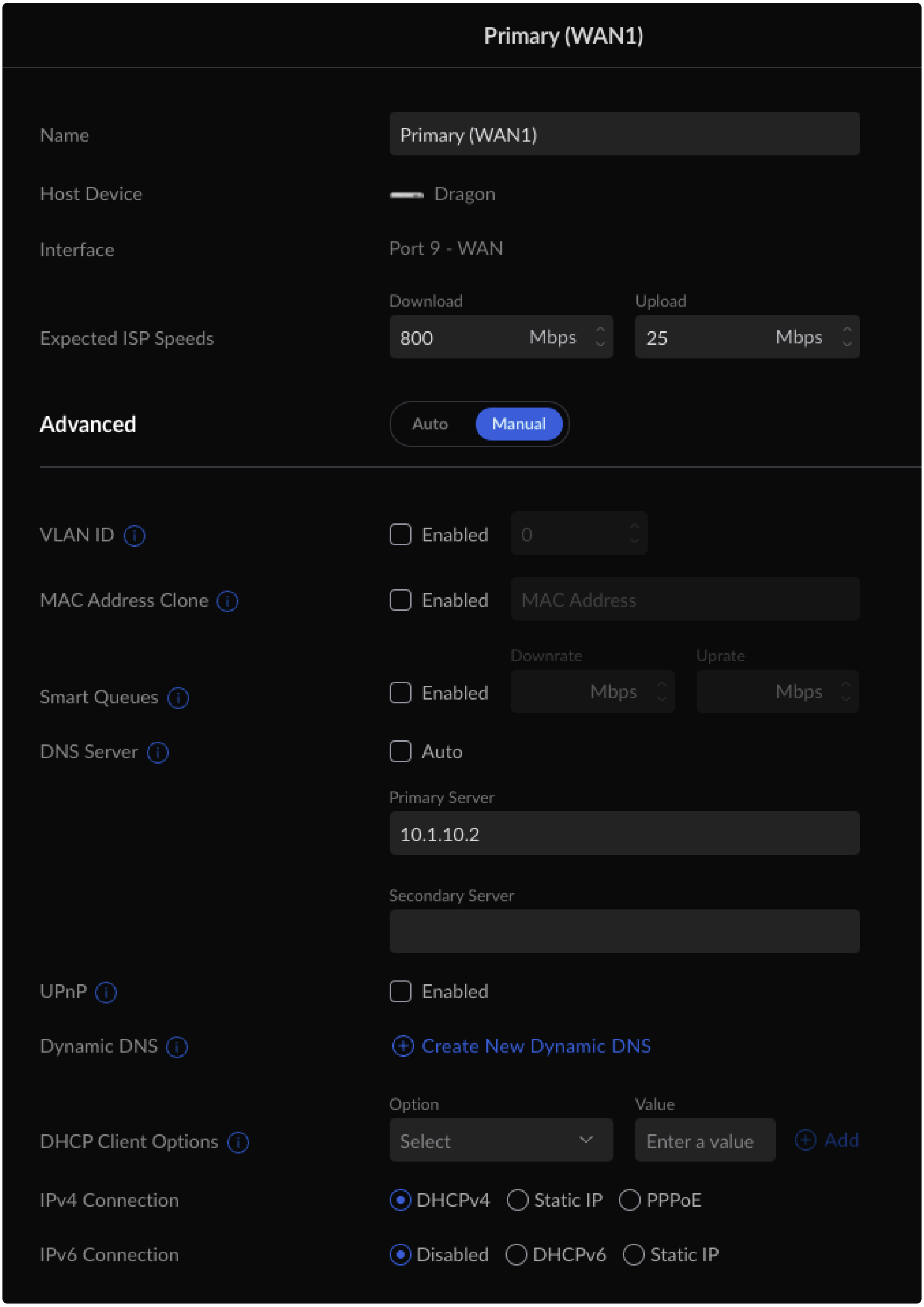
5. Login to PiHole and navigate to **Settings → DNS** uncheck all **Upstream DNS Servers**.

6. Click **Custom 1 (IPv4)** under **Upstream DNS Servers** and enter **127.0.0.1#5335**.

7. Now PiHole will direct all DNS queries through Unbound.

UDM DNS Settings

1. Now that PiHole with Unbound is setup you can set your UDM to direct DNS queries to PiHole. Navigate to **Network → Internet → Primary WAN 1** and set the DNS server to **10.1.10.2**



Primary (WAN1)

Name	Primary (WAN1)
Host Device	Dragon
Interface	Port 9 - WAN
Expected ISP Speeds	Download: 800 Mbps Upload: 25 Mbps
Advanced	Auto / [✓ Manual]
VLAN ID	□ Enabled 0
MAC Address Clone	□ Enabled MAC Address
Smart Queues	□ Enabled Downrate: Mbps Uprate: Mbps
DNS Server	□ Auto Primary Server: 10.1.10.2 Secondary Server:
UPnP	□ Enabled
Dynamic DNS	Create New Dynamic DNS
DHCP Client Options	Option: Select Value: Enter a value Add
IPv4 Connection	● DHCPv4 ○ Static IP ○ PPPoE
IPv6 Connection	● Disabled ○ DHCPv6 ○ Static IP

PiHole Upgrade Process

1. **machinectl shell debian-custom**

2. To update PiHole, simply run the following from within the container: **PIHOLE_SKIP_OS_CHECK=true pihole -up**

3. In case there is a configuration error and PiHole is having trouble, you can reconfigure it from scratch by running: **PIHOLE_SKIP_OS_CHECK=true pihole -r**

Update Ad Block Lists

1. Login to PiHole

2. Navigate to **Adlists**

3. Add
   a. https://big.oisd.nl/
   b. https://raw.githubusercontent.com/d3ward/toolz/master/src/d3host.txt

4. SSH to UDM Pro and run **machinectl shell debian-custom**

5. **pihole -g** (This will update the adblock list)

6. Navigate to ADBlock to test blocking capability.