Install VDO.Ninja

https://www.youtube.com/embed/ND1zMn1uI38

What is VDO.Ninja?

Pronounced 'video ninja'. it's an incredibly powerful real-time video meeting system that will create Peer-to-peer connections when there are only 2 or three people in a call, but utilize a room style connection as the meeting grows. This helps with performance, and bandwidth usage based on the number of attendees.

In this case, you can share your direct VDO.Ninja URL with a friend, family member or colleague, and they can share theirs with you, and you'll have a direct peer-to-peer conversation with great quality and low latency. If, however, you want to have more than a couple of people on a meeting of some kind, then you (the director) can setup a room for a meeting. Share that URL with your invitees, and set a ton of great preferences. You'll have a view where you can be the literal director of the meeting, and control everyone's ability to speak, join, remain, and so on in the call. You can even control the layout of the video squares.

It's truly an awesome, and powerful meeting platform.

Installation of the Server

What you'll need

- A server to host Vdo.Ninja on
- Docker-CE and Docker Compose installed on the server
- A Domain / Subdomain that you can set an A-record for.
- Access to open ports to the server for WebRTC communication if needed
- About 20 minutes of your time.

Install Docker-CE and Docker Compose

Fortunately, the good folks over at docker.com make this a very easy thing to get done. YOu can run this one liner script to get them both installed in most cases:

```
curl -fsSL https://get.docker.com | sh
```

Allow the installation to complete, then we'll add our user to the docker group.

NOTE: You should be using a non-root user who has sudo privileges for any type of server running on the internet.

```
sudo usermod -aG docker <username>
```

Replace <username> with your actual username.

Now, you can make these group changes take effect in one of two ways.

1. Log out and back into the system.
2. use the two commands `newgrp` and `newgrp docker` in that order to temporarily gain access to the docker commands until you log out and back in next time.

Install VDO.Ninja with Docker Compose

1. Create a new folder for the VDO.ninja compose.yaml file to be in.

```
mkdir -p docker/vdoninja
```

2. Now move into that directory with

```
cd docker/vdoninja
```

3. Next, we need to create a new docker compose file and paste in the contents below.

```
nano compose.yaml
```

4. Now paste in these contents

```
---
services:
  vdo.ninja:
    stdin_open: true
    tty: true
    ports:
```

```
    - 80:80
    - 443:443
  restart: unless_stopped
  container_name: vdo.ninja
  environment:
    - SERVER_URL=https://vdo.your-great-domain.com
    - EMAIL_ADDRESS=your@email.com
  image: umlat/vdo.ninja
```

You may need to change the ports 80 and 443 on the left side of the colon, as these refer to incoming ports on your host machine. If ports 80 and 443 are already in use on your machine, then feel free to change the left side port numbers to some other port, e.g.

```
ports:
  - 8042:80
  - 21443:443
```

This is kind of like port forwarding. The left side is the host port, and the right side is the port inside the Docker VM. You are forwarding traffic from the host to the docker VM if the traffic comes in on the port listed on the left side of the mapping.

Now save the file with CTRL + O, and press Enter to confirm, then exit the nano editor with CTRL + X.

Next, we'll pull our images, and start up our containers. I like to pull the images first these days, just because it will elt me know if the yaml syntax is correct inside my compose.yaml file.

```
docker compose pull
```

If that all works properly, move on to start the containers,s but if you encounter an error, read it, and just know there's likely a spacing error or a missing colon in the yaml somewhere, so double check what you've entered.

Now let's start our containers, and monitor their startup with two commands concatenated on a single line:

```
docker compose up -d && docker compose logs -f
```

The first portion `docker compose up -d` tells docker to start the container(s) and run them in `d` etached mode (essentially keep them running in the background for us. The second portion says when the containers start, we want to `f`ollow the logs.

Once the system is up and running, and you see no errors in the logs, you can stop following the logs with the hotkey combo of CTRL + C.

Next, we need to setup a reverse proxy (if you intend to run this on a machine behind a firewall inside your LAN). If you are running this like I did on Digital Ocean, or some other VPS that provides a publicly accessible IPv4 address, then you can simply create an A-record in your DNS to point your FQDN to this IP address on the port you set on the left side of the mapping in the compose.yaml file.

e.g. RECORD TYPE = A --> vdo.yourgreatdomain.com --> 21.22.23.27

Move to your favorite modern browser, and navigate to your domain name and the port number using https. It's important that you access this service through https with valid certificates

Setup a Reverse Proxy

If you are running inside your LAN (or any LAN) and the machine you are hosting VDO.Ninja on does not have a public IPv4 address, then you have two options for passtraffic to that host machine. Both options involve a Reverse Proxy.

Option 1: You port forward ports 80 and 443 to the host machine where you are running the reverse proxy (in my case I use NGinX Proxy Manager), and you setup the Proxy Host entry to point to the Private LAN IP of your host machine, and the port number you specified in the compose.yaml file. Then use that same proxy entry to enable LetsEncrypt for valid SSL certificates.

Option 2 (preferred0: Leave your firewall completely closed off tothe outside world, and instead run a solution using Wireguard (like Netbird.io) on a cloud VPS, then add the client to your VDO.Ninja host. So now you have your host machine reaching out from inside your network to your Reverse Proxy host on the open internet without having any incoming ports open. This is known as "tunneling". You essentially made a special tunnel out through your incoming firewall for your VPN to reach back into your network. Again, setup your Proxy Host to point to your host machines VPN private IP using the port number you setup in the compose.yaml file.

Steps:

1. Make sure you have NGinX Proxy Manager setup and running (i Have docs and video here)
2. Login to NGinX Proxy Manager.
3. Select 'Proxy Hosts' from the dashboard.
4. Click the 'Add New Proxy Host' button in the upper right.
5. In the pop up window enter the FQDN you want for your VDO.Ninja site (keep in mind you must own the domain you are setting up for this, and have an A Record pointing to the host where NGinX Proxy Manager is running for that domain or sub-domain in order for traffic to make it to your proxy host). After typing in the FQDN, press `Tab` to turn the entry into a chip.
6. Move to the IP Address / Hostname field, and enter the private IP (VPN IP) address of your VDO.Ninja host machine.

7. Move to the port field, and enter the port on the left side of the port mapping in your compose.yaml file mapped to port 80 on the right. IF you set `- 8041:80` then you would enter "8041".
8. Enable the options for 'Block Common Exploits', and 'Websocket Suupport', and move to the 'SSL' tab at the top of the pop-up window.
9. Select 'Request a New Certificate' from the drop down.
10. Enable the options for 'Force SSL', 'HTTP/2 Support', and both HSTS options.
11. Enter your email if it's not already entered, and tick the box to agree to the LetsEncrypt TOS.
12. Click 'Save', and be patient. The pop-up window should just close on it's own without any errors showing. If it does, then you have successfully setup your Reverse Proxy Entry.

Now, you should see your new domain in the list of Proxy hosts. You can click on the domain to try and open it in a new tab.

If everything is setup properly, you will be shown a page which looks just like the official VDO.Ninja page.

If you see errors, you will want to go back and check your setup. The flow from the time you enter the URL on the browser, to get to the page will be as follows, so check each step along the way.

Enter the URL --> DNS checks for an A / CNAME record for the FQDN of the URL (vdoninja.mygreatdomain.com) and need it to point to the public IP v4 address where your Reverse proxy is running, or where VDO.Ninja is running if on a VPS

If using a Reverse proxy inside a LAN, once the DNS entry sends it along to your public address, the firewall must have ports 80 and 443 forwarded to the host machine where your reverse proxy is running.

Next, the port forwarding will send the request to your reverse proxy, the reverse proxy will look for that FQDN in it's list, and then send you to the private LAN IP and port of the host machine where VDO.Ninja is running.

If all has gone according to plan, you should be up and running with your own VDO.Ninja server.

A few common issues to check into:

1. Some people's ISPs block ports 80 and 443, so you won't be able to run a Reverse proxy inside your LAN, and the VPN on a VPS approach is the best method (it's really the best method regardless IMO).
2. Some people's ISPs have them assigned only a Private IP, which is known as CG-NAT or Double-NAT, and again you can't reach your LAN because of this. The VPN on a VPS is also the best approach for this situation (IMO).
3. Some firewalls will not port forward for a domain that you have not 'registered' on the firewall itself (e.g. OPNSense, pfSense, etc).

Support My Channel and Content

Support my Channel and ongoing efforts through Patreon:

https://www.patreon.com/awesomeopensource

Buy me a Beer / Coffee:

https://paypal.me/BrianMcGonagill

Revision #3
Created 2025-02-18 13:53:36 UTC by Brian McGonagill
Updated 2025-02-20 12:36:34 UTC by Brian McGonagill