Install and Configure OpenRPort RMM

What Is It?

OpenRPort is an RMM (Remote Machine Management) tool. It's a server based system where you install a client access application on each client machine you want to manage. That client then connects to the server, and becomes accessible in many ways.

What Can I Do With It?

You can view the status and metrics of the client machines / servers.

- CPU Usage
- RAM Usage
- Disk Usage
- Network Usage
- Historical Graphs of I/O and Machine Diagnostic Information
- Audits of actions taken on the client machines
- ...and more

You can also create tunnels to these machines for connection via SSH, RDP, VNC, HTTP, HTTPS, Custom Service connections, and Service Forwarding connections.

These tunnels can be set to self-destroy after a certain amount of time, or inactivity, which lends to additional security.

Users can be limited to only working with certain clients or client groups.

What Do I Need?

- A server to run the OpenRPort Server software on. (they specifically state an LTS - Long Term Support version)
- Curl, Git, wget, and openssh-server installed on yoru host
- About 10 minutes time.
- Access to your Router / Firewall in order to forward ports 80, 443, 5000, 8000, 20000 - 20050 to your host machine. (only necessary if running from inside a LAN and using an

FQDN for access from outside the local network)
- (optional) A VPN for clients to interact with server

> NOTE: you may need to forward 80 and 443 to your reverse proxy if you are running one.

- (Optional) A FQDN (Fully Qualified Domain Name / URL).
  - This can be your own domain / subdomain, or
  - a dynamic DNS service (e.g. duckdns, dyndns, cloudflare ddns, etc).
- (Optional) A reverse proxy (e.g. NGinX Proxy Manager, Traefik, Caddy, HAProxy, etc.), and familiarity with proxy configuration.

Installing the OpenRPort Server

Log into your host machine (preferably with a non-root user that has sudo privileges).

Download the Installer script for your server. **I highly recommend checking the official OpenRPort documentation for any updated scripts / instructions!**

You can use the first set of commands on this page to run inside your home network, like I will do in the video and the rest of these instructions, or you can use the second script on this page to run in a VPS like Digital Ocean, or Linode.

First, download the software from OpenRPort:

```
curl -o rportd-installer.sh https://get.openrport.io
```

If you don't have `curl` installed, you can install it on your system via the package manager of your chosen OS. For Ubuntu / Debian install it with:

```
sudo apt install curl -y
```

I like to just install `wget`, `openssh-server`, `nano`, and `git` as well, just to make it easy. If you want to do that too, you can do it all in one command with:

```
sudo apt install curl wget git nano openssh-server -y
```

After this is done, use the command above to download the OpenRPort server software install script. Here's the command again for convenience:

```
curl -o rportd-installer.sh https://get.openrport.io
```

Now, we need to run the script, and provide a few flags (arguments) in order to make sure we get everything running the way we want.

> PLEASE READ THE ENTIRE SECTION BELOW BEFORE JUST COPY / PASTING THE SNIPPETS

```
sudo bash rportd-installer.sh \
 --email user@example.com \
 --client-port 8000 \
 --api-port 5000 \
 --fqdn rport.localnet \
 --port-range 20000-20050
```

The above command is directly from the OpenRPort site, but I modified mine to change the 2-factor authentication method from "email" to "totp", so I can use BitWarden to get my One Time Pin instead of email. I also changed the FQDN (Fully Qualified Domain Name), to be one on a domain I own, so I can access the OpenRPort server from outside my network (as well, clients from outside can connect to my openrport server).

Below is my version:

```
sudo bash rportd-installer.sh \
 --totp \
 --client-port 8000 \
 --api-port 5000 \
 --fqdn rport.yourcooldomain.org \
 --port-range 20000-20050
```

In the above we specify that we want to authenticate via TOTP (Time-based One Time Passcode), and our url will be `https://rport.yourcooldomain.org`. Of course, change the domain to your own domain / sub-domain.

Now, we can run this command, and OpenRPort server will be installed an started on our host.

> IMPORTANT!
>
> Make note of the URL, username, and password shown in the command line after the server is up and running. You'll need these in order to login for the first time.
>
> Also, have your TOTP app ready, whether that's Bitwarden, Google Authenticator, FreeOTP, and so on.

We need an A Record, or a CNAME Reocrd in our DNS.

In order for the URL we setup to point to our server, we need to setup a special DNS record to point to our public IP address. If you aren't sure what your public IP is, you can go to http://ipchicken.com to see what it is.

Once you have that, you'll want to go to your domain registrar's DNS settings page, and add an A Record DNS entry. Make sure to enter your public IP in the IP address field, and enter the sub-domain (the first part before the primary domain) in the "points to" field.

Domains are generally formatted:

`sub-domain.primary-domain.domain-extension`

so the domain `billsboxes.boxcentral.nu` would be

sub-domain: "billsboxes"
primary-domain: "boxcentral"
domain-extension: "nu"

I can't tell you specifically how your registrar's DNS settings look, or may be worded, as there are dozens of domain registrars, and they can all do things their own way. They should be very close to what I've described above though.

Fixing Your Reverse Proxy

Many people reported that they would get a `502 Bad Gateway` error when setting up Nginx Proxy Manager as the reverse proxy using the method above. The fix for that is below. You still need to setup your reverse proxy, but if you are using NginX Proxy Manager like I do, you need to do a couple of extra steps.

First, log into your OpenRPort server.

Next, edit the file `/etc/rport/rportd.conf` using your favorite text editor. I'll use nano, so the command is:

`sudo nano /etc/rport/rportd.conf`

```
## If both cert_file and key_file are specified, then rportd will use them to serve the API
with https.
## Intermediate certificates should be included in cert_file if required.
cert_file = "/etc/rport/ssl/rpt.routemehome.org_certificate.pem"
key_file = "/etc/rport/ssl/rpt.routemehome.org_privkey.pem"
```

Remote the paths for both the "cert_file" and the "key_file". After you're done, it should look like:

```
## If both cert_file and key_file are specified, then rportd will use them to serve the API
with https.
## Intermediate certificates should be included in cert_file if required.
cert_file = ""
key_file = ""
```

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

Reboot your server.

Now, in the reverse proxy, set it up with `http` instead of `https` on the first tab. Also, in the SSL tab enable Force SSL, HTTP/2 Support, and HSTS.

Save, and you should be set and ready to go.

> Thanks to @Yellowfever in my YouTube comments for this fix.

Continuing with Setup

Now, if you have a reverse proxy (like I do), then you'll want to make sure ports 80 and 443 are forwarded from the outside internet, through your router / firewall, to your proxy host. If you aren't using a reverse proxy, then you want to forward those ports directly to your OpenRPort server's internal IP address (usually something like 192.168.x.x, or 10.x.x.x, or 172.x.x.x).

In your reverse proxy, you'll want to proxy any request for https://yourrport.yourcooldomain.org (using your domain, of course) on port 5000 to your server for the Web UI access. You'll want to create an SSL certificate, but a self-signed certificate is also created.

Additionally, regardless of whether you run a reverse proxy, you'll want to forward the ports 8000, 20000 - 20050 to your OpenRPort Server's private IP.

Now, we've got all of our routing setup. Hopefully, it's all set correctly.

You should now be able to navigate in a modern browser to your domain / subdomain.

"https://yourrport.yourcooldomain.org" , and you shouldn't have to add the 5000 port.

You may be greeted with a server certificate warning. Feel free to accept the risk, as this is your server.

Next, you'll want to use the username / password combination provided in the terminal when you created the server, to login for the first time. Also, if you set the TOTP option, make sure to have a TOTP app ready to scan the QR code during setup.

A More Secure Setup

If you are considering utilizing OpenRPort for business, it's always wise to think about better security. As I've stated above, it's not required to use an FQDN with OpenRPort. You can use just

the IP address on the private LAN for the host. This also applies to using a VPN private address. The benefit to using a VPN address between your OpenRPort Host, and any of the machines that need to communicate with it, is that you can keep all the traffic inside that Virtual Private Network, and keep it securely encrypted.

When dealing with clients, their machines, their data, and access to those machines and data, it is of the utmost importance to take every precaution you can to keep their machine access and data secure.

The best thing about this, is there is no requirement of a proxy. We can access everything via our VPN. The is no need for port forwarding either. In fact, we only need a proxy if we want to access our Web UI from a machine not on our VPN (which we really shouldn't do, ever, as we never know what's running on someone else's machine).

The VPN Setup

I'll be using my Netbird VPN for this setup, but feel free to use any VPN you are familiar with. There are several ways this type of setup can work, but the best, is of course via Peer-to-Peer connectivity, which is exactly what we can get with a Wireguard based VPN system like Netbird.

1. First, let's add our new OpenRPort server to our VPN. Since our server is running headless (no GUI Desktop) we'll use the setup keys to do this.
2. Let's add our OpenRPort server to a new Group.
    1. Really we want to add it to every group we create for each of our clients (client businesses). So, if you server 4 businesses, Bills Rentals, Cup 'O Joe, Lit, and LocoMotive. In Netbird, you should create a group per Business. You don't have to just use a single group per business. Depending on their needs and devices, it could make sense to make several groups per business. Let's just stick with 1 group per business for now. What I'll do is create a fifth group called SysMainIT (my business group for my own machines, servers, montiros, etc).
        1. Groups:
            1. BillsRentals
            2. CupOJoe
            3. Lit
            4. LocoMotion
            5. SysMainIT
3. Now, we can add machines for each business to our VPN, and make sure to add the machine to the proper group(s).
4. We add our OpenRPort server to each of these 5 groups.
5. Turn off 'All' for the group routing rules, and create new rules where each group of machines can only communicate within it's own group.
6. Now, we can use the VPN based IP address of our OpenRPort server for each client to reach out to it.

Initial Settings

Once you log in for the first time, you'll want to head to the settings (gear icon) and set the General system settings to your liking.

You'll also want to change the password for the admin user from the password generated during install. You can now add more users, and start adding client machines.

Check out the video to see how to add clients, and more great info on the OpenRPort user interface.

Revision #1
Created 2024-10-06 18:08:49 UTC by Brian McGonagill
Updated 2024-10-06 18:36:59 UTC by Brian McGonagill