OpenVPN Bridges

Prepare OpenVPN

- Rather than repeating the already excellent documentationf from Ubuntu, we are just going to give you the URL with the instructions to follow in order to get a OpenVPN server installed and configured.
- https://ubuntu.com/server/docs/service-openvpn
- You can complete the steps up to and including **Simple Server Configuration**.
- After this we will remove the **/etc/openvpn/server.conf** file and replace it with the following three files.
- Please adapt to your server, especially the value of **local** to reflect the public IP Address of your server.
- We are going to use OpenVPN configured as follows:
  - We are not going to use encryption of the tunnel.
  - We are not going to use the PKI.
  - We are not compressing the data.
  - We are using a username and password given by the client and pass it onto a script to verify if the client is valid.
- Remove /etc/openvpn/server.conf

```
sudo rm /etc/openvpn/server.conf
```

OpenVPN server config for br0.101

- Create a file called **/etc/openvpn/server_vlan_101.conf**.
- Be sure to check the correct value for eth1.101. It might be eth0.101 with your config. (up "/etc/openvpn/up.sh br0.101 eth0.101")

server_vlan_101.conf

```
mode server

auth none

tmp-dir /dev/shm

auth-user-pass-verify "/etc/openvpn/openvpn_auth.pl" via-file
verify-client-cert none
username-as-common-name
script-security 2

local 178.1.1.20
port 1194
proto udp
dev tap
ca ca.crt
cert server.crt
key server.key  # This file should be kept secret
```

```
dh dh.pem

up "/etc/openvpn/up.sh br0.101 eth1.101"
server-bridge 10.101.0.1 255.255.0.0 10.101.0.2 10.101.0.100

ifconfig-pool-persist ipp.txt
;client-config-dir ccd
keepalive 10 120
persist-key
persist-tun
status openvpn-status.log
verb 4
allow-compression no
data-ciphers none
cipher none
```

OpenVPN server config for br0.102

- Create a file called **/etc/openvpn/server_vlan_102.conf**.
- Be sure to check the correct value for eth0.102. It might be eth1.102 with your config. (up “/etc/openvpn/up.sh br0.102 eth1.102”)

server_vlan_102.conf

```
mode server

auth none

tmp-dir /dev/shm

auth-user-pass-verify "/etc/openvpn/openvpn_auth.pl" via-file
verify-client-cert none
username-as-common-name
script-security 2

local 178.1.1.20
port 1195
proto udp
dev tap
ca ca.crt
cert server.crt
key server.key  # This file should be kept secret
dh dh.pem

up "/etc/openvpn/up.sh br0.102 eth1.102"
server-bridge 10.102.0.1 255.255.0.0 10.102.0.2 10.102.0.100

ifconfig-pool-persist ipp.txt
;client-config-dir ccd
```

```
keepalive 10 120
persist-key
persist-tun
status openvpn-status.log
verb 4
allow-compression no
data-ciphers none
cipher none
```

OpenVPN server config for br0.103

- Create a file called **/etc/openvpn/server_vlan_103.conf**.
- Be sure to check the correct value for eth0.103. It might be eth1.103 with your config. (up “/etc/openvpn/up.sh br0.103 eth1.103”)

server_vlan_103.conf

```
mode server

auth none

tmp-dir /dev/shm

auth-user-pass-verify "/etc/openvpn/openvpn_auth.pl" via-file
verify-client-cert none
username-as-common-name
script-security 2

local 178.1.1.20
port 1196
proto udp
dev tap
ca ca.crt
cert server.crt
key server.key  # This file should be kept secret
dh dh.pem

up "/etc/openvpn/up.sh br0.103 eth1.103"
server-bridge 10.103.0.1 255.255.0.0 10.103.0.2 10.103.0.100

ifconfig-pool-persist ipp.txt
;client-config-dir ccd
keepalive 10 120
persist-key
persist-tun
status openvpn-status.log
verb 4
allow-compression no
data-ciphers none
```

```
cipher none
```

Prepare /etc/openvpn/up.sh

- You'll see in the config files there are reference to two scripts.
- **/etc/openvpn/up.sh** is called when the tap interface comes up.
- **/etc/openvpn/openvpn_auth.pl** is used to verify the clients.
- Create the /etc/openvpn/up.sh file

```
sudo vi /etc/openvpn/up.sh
```

- Give it the following content:

up.sh

```
#!/bin/sh

BR=$1
ETHDEV=$2
TAPDEV=$3

/sbin/ip link set "$TAPDEV" up
/sbin/ip link set "$ETHDEV" promisc on
/sbin/brctl addif $BR $TAPDEV
```

- Make it executable

```
sudo chmod 755 /etc/openvpn/up.sh
```

Prepare openvpn_auth.pl

- You can run the OpenVPN server separate from the RADIUSdesk server.
- The **openvpn_auth.pl** script can then be copied to the server running the OpenVPN server.
- You just have to configure the **openvpn_auth.pl** script to point to your RADIUSdesk server to do the API calls when authenticating a client.
- The **openvpn_auth.pl** script is traditionally under **/var/www/rdcore/cake4/rd_cake/setup/scripts/**.
- Copy this file to **/etc/openvpn/** on the OpenVPN server and edit the following to point to your RADIUSdesk server.

```
my $protocol='http';
my $server_name_or_ip='198.27.111.78';
my $api_path="/cake4/rd_cake/openvpn-servers/auth-client.json";
```

- Make sure this file is executable in its new location:

```
sudo chmod 755 /etc/openvpn/openvpn_auth.pl
```

- Make sure the Perl modules that are used by this script are installed.

```
sudo apt-get install liblwp-protocol-https-perl
```

Test Start OpenVPN service

- Be aware that the **systemctl start openvpn** is not starting your openvpn you just defined.
- Openvpn uses templatized systemd jobs, openvpn@CONFIGFILENAME.
- So if for example your configuration file is myserver.conf your service is called openvpn@myserver.
- You can run all kinds of service and systemctl commands like start/stop/enable/disable/preset against a templatized service like openvpn@server.
- With that in view you can tests start the individual servers based on their config files.

```
#start 101
sudo systemctl start openvpn@server_vlan_101
#check the output for any errors
journalctl -xeu openvpn@server_vlan_101.service
#start 102
sudo systemctl start openvpn@server_vlan_102
#check the output for any errors
journalctl -xeu openvpn@server_vlan_102.service
#start 103
sudo systemctl start openvpn@server_vlan_103
#check the output for any errors
journalctl -xeu openvpn@server_vlan_103.service
```

Check the bridges

- Confirm that the bridges each have a tap interface included:

```
brctl show
bridge name bridge id         STP enabled    interfaces
br0.101     8000.000c294aafdf    no         eth0.101
                            tap0
br0.102     8000.000c294aafdf    no         eth0.102
                            tap1
br0.103     8000.000c294aafdf    no         eth0.103
                            tap2
```

Install ifconfig

- Although ifconfig does not come standard with recent versions of Ubuntu and we try to use the replacement **ip** command where possible, we will install ifconfig to help here.
- It is also required during the next section when we work with CoovaChilli.

```
sudo apt install net-tools
```

Test ifconfig

- ifconfig should also include a list of three tap interfaces

```
tap0      Link encap:Ethernet  HWaddr 22:1a:35:b6:01:d7
          inet6 addr: fe80::201a:35ff:feb6:1d7/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:16 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:768 (768.0 B)  TX bytes:820 (820.0 B)

tap1      Link encap:Ethernet  HWaddr ca:e0:7d:c0:ea:a0
          inet6 addr: fe80::c8e0:7dff:fec0:eaa0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 B)  TX bytes:648 (648.0 B)

tap2      Link encap:Ethernet  HWaddr f2:36:e7:d2:da:c1
          inet6 addr: fe80::f036:e7ff:fed2:dac1/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 B)  TX bytes:648 (648.0 B)
```

OpenVPN Startup

- Since we use our own startup script to prepare the environment we will disable the normal startup sequience of OpenVPN.
- We will start it up **after** our environment has been prepared.
- Disable the startup of OpenVPN

```
sudo systemctl disable openvpn
```

- Add the following lines to the **/etc/rc.local** file to ensure OpenVPN starts up **after** the bridges have been set up.

```
/sbin/ip addr add 10.103.0.1/16 dev br0.103
/sbin/ip link set dev br0.103 up

#Add the startup of OpenVPN
systemctl start openvpn@server_vlan_101
systemctl start openvpn@server_vlan_102
systemctl start openvpn@server_vlan_103

exit 0
```

- We are making good progress. Next we will install and configure **Coova Chilli** so that it runs an instance on each VLAN.

- To confirm everything will come up after a power cycle, go ahead and reboot the server.

```
sudo reboot
```