Networking

Linux network troubleshooting and administration

- CentOS 7 - Configuring Cacti
- IPTables - Forwarding Between LAN and WLAN
- Connecting to OpenVPN Using NetworkManager
- OpenVPN - Firewall Configuration
- RHEL VLAN and Bonding Configuration
- Setup SFTP to Public Directory (/var/www)
- FirewallD - Opening a Port
- Using LetsEncrypt for OpenVPN WebSSL
- Monitor Mode on Broadcom-wl Driver
- SSH Tunneling
- Syncing Files with FTP
- ArchLinux - Setting Up Fortinet SSL VPN

CentOS 7 - Configuring Cacti

Install Required Dependencies

```
yum -y install mariadb-server php php-cli php-mysql net-snmp-utils rrdtool php-snmp gcc mariadb-
```

Enable Required Services for Cacti

```
chkconfig httpd on
chkconfig mariadb on
chkconfig crond on
```

Download and Extract Cacti

cd /var/www/html
wget http://www.cacti.net/downloads/cacti-0.8.8c.tar.gz
tar -xzvf cacti-0.8.8c.tar.gz

Setting Up Cacti for Apache

Add Cacti User & Enable Cron Jobs

```
adduser cacti
echo "*/5 * * * * cacti php /var/www/html/cacti/poller.php &>/dev/null" >> /etc/cron.d/cacti
```

Fix Cacti Directory Permission

```
cd /var/www/html/cacti
chown -R cacti.apache rra log
chmod 775 rra log
```

Set Up Cacti Database

```
mysql -p cacti < /var/www/html/cacti/cacti.sql
GRANT ALL ON cacti.* TO cactiuser@localhost IDENTIFIED BY 'MyV3ryStr0ngPassword';
flush privileges;
exit
cd /var/www/html/cacti/include/
vi config.php (and change $database_* configuration and $url_path)
```

Open Firewall Ports to HTTP and HTTPS

```
firewall-cmd --permanent --zone=public --add-service=https
firewall-cmd --permanent --zone=public --add-service=http
firewall-cmd --reload
```

Login to cacti using admin:admin and go to “Console -> System Utilities” and click on “Rebuild Poller Cache” after the first login!

IPTables - Forwarding Between LAN and WLAN

Add the following to `/etc/udev/rules.d/10-network.rules`, substitute `LAN_MAC_ADDR` and `WLAN_MAC_ADDR` with your Ethernet device and WLAN device MAC addresses for persistent network names:

```
SUBSYSTEM=="net", ACTION=="add", ATTR{address}=="LAN_MAC_ADDR", NAME="ether0"
SUBSYSTEM=="net", ACTION=="add", ATTR{address}=="WLAN_MAC_ADDR", NAME="wifi0"
```

Add the following to `/etc/sysctl.d/30-ip_forward.conf`:

```
net.ipv4.ip_forward=1
net.ipv4.conf.default.forwarding=1
net.ipv4.conf.all.forwarding=1
Add the following to /etc/iptables/iptables.rules:
*nat
:PREROUTING ACCEPT [783:65928]
:INPUT ACCEPT [73:9660]
:OUTPUT ACCEPT [6180:382480]
:POSTROUTING ACCEPT [18:1260]
-A POSTROUTING -o wifi0 -j MASQUERADE
COMMIT

*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [176:192839]
-A INPUT -i lo -m comment --comment "Inbound from loopback (lo)" -j ACCEPT
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -j NFLOG --nflog-group 1
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -i wifi0 -j ACCEPT
-A FORWARD -i wifi0 -o ether0 -m comment --comment "ether0 <\- wifi0" -j ACCEPT
-A FORWARD -i ether0 -o wifi0 -m comment --comment "wifi0 -> ether0" -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

Connecting to OpenVPN Using NetworkManager

Install the required packages

```
sudo apt-get install network-manager network-manager-openvpn network-manager-openvpn-gnome
```

Creating individual files from client.ovpn file

These files must be kept safe and private at all times

1. Make a directory called openvpn in your home directory
2. Copy the client.ovpn file into dir openvpn
3. Optional: Keep an original copy of the file – call it client.ovpn.orig
4. Next we will create 4 files under the openvpn directory. Open the client.ovpn file in a text editor
5. Create a file called ca.crt – copy the text between and from client.ovpn into this file
6. Create a file called client.crt – copy the text between and from client.ovpn into this file
7. Create a file called client.key – copy the text between and from client.ovpn into this file
8. Create a file called ta.key – copy the text between and from client.ovpn into this file
9. At this point i have a total of 6 files under my openvpn directory

Modify the client.ovpn file

Just before the ## —–BEGIN RSA SIGNATURE—– line add the below lines and save:

```
ca ca.crt
cert client.crt
key client.key
tls-auth ta.key
```

Setting up the Network Manager

1. Click on Ubuntu network icon on the top right
2. Select VPN Connections -> Configure VPN (the Network Connections window will open)
3. Click on the VPN tab and click Import
4. Select the client.ovpn file we just modified and it should automatically import some things into the next screen
5. Connection Name will be = client – change this to something meaningful (i set it to companyVPN)
6. Gateway must be imported already
7. Type is : Password with Certificates (TLS) – this was also set for me
8. Provide the username and password for VPN
9. User certificate will be client.crt
10. CA certificate will be ca.crt
11. Private Key will be client.key
12. Click on Advanced -> TLS Authentication Tab
13. Key file will be ta.key
14. Key Direction must be set based on the key direction in your client.ovpn file
15. Open the client.ovpn file and search for “key-direction” and note the number after that (mine is key-direction 1)
16. Put this number in the Key Direction field in the TLS Authentication Tab
17. Click save on all windows and close all windows.

Time to test connection

1. Click on network icon on the top right
2. Select VPN Connections and you should see your connection there – click it
3. If successfully connected, you will see a message and then you can verify your IP address with ifconfig
4. There is a Disconnect VPN under VPN Connection for obvious reasons

OpenVPN - Firewall Configuration

FirewallD

Use the following commands to open all ports required by OpenVPN:

```
firewall-cmd --list-services
firewall-cmd --permanent --add-service openvpn
firewall-cmd --permanent --add-masquerade
firewall-cmd --query-masquerade
firewall-cmd --reload
```

IPTables

My IPTables configuration `/etc/iptables/iptables.rules` for OpenVPN:

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [32:2712]
:LOGGING - [0:0]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -m icmp --icmp-type 8 -j REJECT --reject-with icmp-host-prohibited
-A INPUT -p icmp -m icmp --icmp-type 0 -j REJECT --reject-with icmp-host-prohibited
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo0 -m comment --comment "Allow loopback lo0" -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT
-A INPUT -p udp -m udp --dport 1194 -j ACCEPT
-A INPUT -j LOGGING
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -i tun+ -j ACCEPT
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
-A LOGGING -j LOG --log-prefix "DROPPED: " \--log-level 7
-A LOGGING -j DROP
COMMIT
# Completed on Mon Jun 30 06:48:44 2014
# Generated by iptables-save v1.4.7 on Mon Jun 30 06:48:44 2014
```

```
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [2:165]
:OUTPUT ACCEPT [2:165]
-A POSTROUTING -s 10.8.0.0/24 -o eth0 -j MASQUERADE
COMMIT
```

RHEL VLAN and Bonding Configuration

Check list:

- Check whether the 8021q module has been loaded.
- lsmod | grep 8021q
- If the 8021q module is not loaded, run the following command to load it: modprobe 8021q

Configuration

Add the following lines to /etc/modprobe.conf :

```
alias bond0 bonding

options bonding max_bonds=1

Edit /etc/sysconfig/network-scripts/ifcfg-eth0 it should look something like this:
DEVICE=eth0
USERCTL=no
ONBOOT=yes
MASTER=bond0
SLAVE=yes
BOOTPROTO=none
HWADDR=

Edit /etc/sysconfig/network-scripts/ifcfg-eth1 it should look something like this:
DEVICE=eth1
USERCTL=no
ONBOOT=yes
MASTER=bond0
SLAVE=yes
BOOTPROTO=none
HWADDR=
```

Now create the Bond0 interface:

NOTE: No IP address will be assigned to the bond0 device.

Create a new file `/etc/sysconfig/network-scripts/ifcfg-bond0` it should look like this:

```
DEVICE=bond0
BOOTPROTO=none
ONBOOT=yes
TYPE=Ethernet

BONDING_OPTS="mode=1 miimon=100"
```

NOTE: mode could be different, these are the mode options, but if Blade server is using Virtual Connect user should use mode=1.

mode=0 (balance-rr) Round-robin
mode=1 (active-backup) Active-backup
mode=2 (balance-xor) XOR
mode=3 (broadcast) Broadcast
mode=4 (802.3ad) IEEE 802.3ad Dynamic link aggregation
mode=5 (balance-tlb) Adaptive transmit load balancing
mode=6 (balance-alb) Adaptive load balancing

The first four modes are the most commonly used:

VLAN tag setup

This will be a virtual interface with a VLAN tag of 48. User's VLAN set-up is most likely different so just replace 48 with the VLAN tag of user's network. i.e. bond1.50 would be the bonded interface for VLAN 50.

Create a new file `/etc/sysconfig/network-scripts/ifcfg-bond0.48` it should look like this:

```
DEVICE=bond0.48
ONBOOT=yes
TYPE=Ethernet
BOOTPROTO=static
VLAN=yes
NETMASK=255.255.255.0
NETWORK=192.168.48.0
IPADDR=192.168.48.100
```

Ensure that the default gateway in this configuration is recorded in the /etc/sysconfig/network file otherwise it may not work properly. Once done, it should look something like:

```
NETWORKING=yes
HOSTNAME=
GATEWAY=192.168.48.1
```

User has now setup bonding and VLAN tagging. User needs to restart networking to make the changes active.

```
service network restart
```

Testing

Verify bonding interface is up and running

```
ifconfig -a
```

Verify configuration (RHEL 5 is using `sysfs` , so check out `/sys/class/net/`)

Setup SFTP to Public Directory (/var/www)

Configuring SSH for SFTP

1. `vim /etc/ssh/sshd_config`
2. Comment the following line:
    1. `Subsystem sftp /usr/local/libexec/sftp-server`
3. Add the following lines:
    1. `Subsystem sftp internal-sftp`
    2. `Match Group <sftp group>`
    3. `ChrootDirectory %h`
    4. `ForceCommand internal-sftp`
    5. `X11Forwarding no`
    6. `AllowTcpForwarding no`
4. Save and close
5. Reload ssh `sudo systemctl restart sshd`

Add SFTP User and Set Permission

1. `sudo groupadd <user> -g <sftp group> -s /bin/false -d /var/www/html`
2. `sudo passwd <user>`
3. `sudo chown root /var/www/html`
4. `sudo chmod 755 /var/www/html`
5. `sudo mkdir /var/www/html/<dir>`
6. `sudo chmod 775 /var/www/html/<dir>`
7. `sudo chown apache: apache /var/www/html/<dir>`
8. `sudo chmod g+s /var/www/html/<dir>`

Selinux

1. `sudo setsebool -P ssh_chroot_rw_homedirs on`
2. `sudo setsebool -P httpd_unified 1`
3. `sudo setfacl -d -m g: apache: rw /var/www/html/<dir>`

References

1. Spiceworks Article
2. CentOS Docs

FirewallD - Opening a Port

Use this command to find your active zone(s):

```
firewall-cmd --get-active-zones
```

It will say either public, dmz, or something else. You should only apply to the zones required.

In the case of dmz try:

```
firewall-cmd --zone=dmz --add-port=<port>/tcp --permanent
```

Otherwise, substitute dmz for your zone, for example, if your zone is public:

```
firewall-cmd --zone=public --add-port=<port>/tcp --permanent
```

Then remember to reload the firewall for changes to take effect.

```
firewall-cmd --reload
```

Using LetsEncrypt for OpenVPN WebSSL

Using letsencrypt for OpenVPN Access Server is nothing more than symlinking the files to letsencrypt keys and certs:

1. `sudo -s`
2. `cd /usr/local/openvpn_as/etc/`
3. `mv web-ssl web-ssl.bak`
4. `mkdir web-ssl`
5. `ln -s /etc/letsencrypt/live/<letsencrypt domain dir>/privkey.pem web-ssl/server.key`
6. `ln -s /etc/letsencrypt/live/<letsencrypt domain dir>/cert.pem web-ssl/server.crt`
7. `ln -s /etc/letsencrypt/live/<letsencrypt domain dir>/fullchain.pem web-ssl/ca.crt`
8. `systemctl restart openvpnas`

Monitor Mode on Broadcom-wl Driver

Enable monitor mode:

```
$ echo 1 > /proc/brcm_monitor0
```

Enabling monitor mode will create a `prism0` network interface. Wireshark and other network tools can use this new `prism0` interface.

To disable monitor mode:

```
$ echo 0 > /proc/brcm_monitor0
```

SSH Tunneling

`ssh -p <port> <username>@<remote host> -L <local listening port>:<remote's host ip>:<remote's host port> -N`

Note:
Remote's host and port can be any host and port accessible by the remote host, e.g. to access the router web interface on 192.168.1.1 (remote) use <local listening port>:192.168.1.1:80

Syncing Files with FTP

I came across a problem when doing migration last time, the server grew too big that I cannot just simply compress the files and move it to another server, it was more than 20GB files. So, I came across an FTP client called LFTP that will synchronize files and folders over FTP. The script below is the script I used to sync the files, let's call it sync.sh:

```
#! /bin/bash
HOST='<ftp host>'
USER='<ftp user>'
PASS='<password>'
RCD='<remote directory to sync>'
lftp -e "
open $HOST
user $USER $PASS
mirror --verbose --continue $RCD
bye
"
```

To sync only certain folders, use the following sciprts:

```
#! /bin/bash
HOST='<ftp host>'
USER='<ftp user>'
PASS='<password>'
RCD='<remote directory to sync>'
lftp -e "
open $HOST
user $USER $PASS
mirror --verbose --continue --exclude '.*' --exclude '.*/' --include '<folder1>' --
include '<folder2>' $RCD
bye
"
```

To have sync overnight even when logged out, use the command `nohup bash sync.sh > sync.log`.

ArchLinux - Setting Up Fortinet SSL VPN

1. Install ppp, openfortivpn and networkmanager's fortinet plugin package: `sudo pacman -Syu ppp openfortivpn networkmanager-fortisslvpn`
2. Get certificate digest by running: `sudo openfortivpn <IP Address>:<Port> --username=<username>`
3. Enable kernel module: `modprobe ppp_generic`
4. Reconnect with openfortivpn: `sudo openfortivpn <IP Address>:<Port> --username=<username> --trusted-cert <certificate digest>`
5. Now you can connect to the VPN by creating a new Fortinet SSLVPN (fortisslvpn) connection:
   - Enter the `Gateway` in the format `<IP Address>:<Port>`
   - Your username and password
   - Finally click "Advanced" and enter the certificate digest into `Trusted certificate` field

References

- ArchLinux PPP
- Openfortivpn