Advance LINUX Notes

Photo by Lukas on Unsplash

Control Service and Daemons

What is the process?

A process is an instance of a particular executable (.exe program file) running. A given application may have several processes running simultaneously.

What is Daemon?

The word “daemon” actually comes from the Greek language, meaning an “inner or attendant spirit” (Oxford American Dictionary). This is a fitting name, as a computer daemon is a constantly running program that triggers actions when it receives certain input.

What is Service?

A service is a program which responds to requests from other programs over some inter-process communication mechanism (usually over a network). A service is what a server provides. For example, the NFS port mapping service is provided as a separate portmap service.

Service Status

1. Active(Running)
2. InActive(Not Running)
3. Enabled(Start at boot)
4. Disabled(Do not start at boot)

1. **Display status of service**

```
#systemctl status sshd
```

2. Restart service

```
#systemctl restart sshd
```

3. Stop Service

```
#systemctl stop sshd
```

4. Start Service

```
#systemctl start sshd
```

5. Permanent on Service

```
#systemctl enable sshd
```

6. Permanent off service

```
#systemctl disable sshd
```

7. Determine service is enabled?

```
#systemctl is-enabled sshd
```

. . .

Package Management

What is Package

All software on a Linux system is divided into packages that can be installed, uninstalled or upgrade. In Linux distributions, a “package”

refers to a compressed file archive containing all of the files that come with a particular application.

Package Architecture

httpd-tools-2.4.6–7.el8.x86_64.rpm

Name version.release architecture extention

Method for install package

1. **Standalone installation-**
   This method use for single or few no of computers.
   Eg. Cd, dvd, pd etc.

2. **Network Installation-**

This method is used for large number of computers.
Eg. Centralized Server

Standalone Installation

rpm : (Red Hat Package Manager)

RPM command is used to install package in Linux like (RHEL, CentOS and Fedora).
rpm command use for standalone installation.

Drawback of rpm command :
1. We Cant install dependent software in single shot
2. It Does not wait for user confirmation.

rpm Command syntax

```
#rpm <option> <name of package>
```

Option:
i -for install
v -for verbose
h -for hashes
e -for erase
q -for query

Install package with rpm

```
#rpm -ivh httpd-tools-2.4.6-7.el7.x86_64.rpm
```

2. Network Installation

Yum: (Yellowdog Updater, Modified)
YUM is an open source package management tool for
install rpm based packages in linux system
Yum overcome all drawback of rpm command.

It allows users and system administrator to easily
install, update, remove or search software packages
on a systems.

Yum Command syntax:

```
#yum <option> <Name of Package>
```

Options :
install -for install
remove -for uninstall
upgrade -for upgrade
groupinstall -for install group of application
groupremove -for uninstall group of application
grouplist -for show list of group
clean all -for clean repository cache
list -for show package list

Install package with yum

```
#yum install httpd*
```

Note :
For install package with yum command we need to set yum path of centralized server in our system (repo)

Steps for create repository file (repo)

Step 1: Insert rhel 8.0 dvd and start VM:

attach RHEL8 iso file in VM.

Step 2: For show DVD mount point

```
#lsblk
```

Step 3: Go to yum repository

```
#cd /etc/yum.repos.d/
#rm -rvf *
```

For create repository file (repo)

Step 5: Create repo file

```
#vim server.repo
```

```
[app]
name=appstrea
baseurl=file:///run/media/root/RHEL-8-0-0-BaseOS-X86_64/AppStream
enabled=1
gpgcheck=0
[base]
name=baseos
baseurl=file:///run/media/root/RHEL-8-0-0-BaseOS-X86_64/BaseOS
enabled=1
gpgcheck=0
```

:wq

Step 6: `#yum clean all`

```
#yum repolist
#yum list
```

For create repository file (repo)

Setup static ip to server and client OS
Eg. 192.169.1.3

Install Web server Package and start enable service

```
#yum install httpd*
#systemctl start httpd
#systemctl enable httpd
```

Copy and paste all package in `/var/www/html/`

```
#cd /run/media/root/RHEL-8-0-0-BaseOS-X86_64/
#cp -rvf AppStream /var/www/html/
#cp -rvf BaseOS /var/www/html/
```

For create repository file (repo)

Create Repo file

```
#cd /etc/yum.repos.d/
#vim server.repo
[app]
name=appstream
baseurl=http://192.168.1.2/AppStream
```

```
enabled=1
gpgcheck=0
[base]
name=baseos
baseurl=http://192.168.1.2/BaseOS
enabled=1
gpgcheck=0
```

Setup Firewall Rule :

```
 #firewall-cmd—permanent --add-service=http
#firewall-cmd—reload
#systemctl restart httpd
```

Configure client side yum repo

```
 #vim /etc/yum.repos.d/
#rm -rvf *

 #vim servers.repo
[app]
name=appstream
baseurl=http://192.168.1.2/AppStream
enabled=1
gpgcheck=0
[base]
name=baseos
baseurl=http://192.168.1.2/BaseOS
enabled=1
gpgcheck=0
```

:wq

```
 #yum clean all
#yum repolist
#yum install http*
```

. . .

Secure Shell—SSH

What is SSH?

- SSH is a protocol used to securely access remote computer by using command
  line interface.

- It use by default port number 22.
- ssh is the most common way to access remote Linux and Unix-like servers.
- In history telnet use for access remote computer command line interface but due to security reason telnet outdated and use ssh today instead of telnet.

Update ssh configuration file

To enable root remote access through remote system modify sshd_config file

To edit ssh configuration file

```
#vim /etc/ssh/sshd_config
PermitRootLogin yes (line number 40)
PasswordAuthentication yes (line number 65)
```

:wq

Remove **#** above line and update as per requirement.

```
#systemctl restart sshd
```

For access remote system with SSH

For Connect remote system:

```
#ssh root@<ip_Address>
```

For Check:

```
#ifconfig
```

For Logout :

```
#exit (ctrl+d)
```

SSH key-based Authentication

For Generate key

```
#ssh-keygen
```

For transfer key to remote system

```
#ssh-copy-id -i <ip_Address>
```

Now,try to login remote system password not required

```
#ssh root@<ip_Address>
```

Disable SSH root user login

Edit configuration file:

```
# vim /etc/ssh/sshd_config
PermitRootLogin no
DenyUsers amir
```

:wq

```
#systemctl restart sshd
#systemctl enable sshd
```

. . .

Remote File Transfer in Linux

File Transfer

In Linux for transfer file local to remote and remote to local use following command popularly.

1. scp
2. rsync

SCP (Secure Copy)

scp is a command line tool for Linux systems for securely transfer files from local to remote server or vice a versa. SCP uses SSH protocol for transferring files between two systems which is more secure than ftp.

Transfer File Local to Remote Server

For File :

```
#scp myfile.txt root@<ip_Address>:/mnt/
```

For Directory :

```
#scp -r /india root@<ip_Address>:/mnt/
```

Transfer file remote server to local

For File :

```
#scp root@<ip_Address>:/mnt/myfile.txt /root/Desktop/
```

For Directory :

```
#scp -r root@<ip_Address>:/ root@<ip_Address>:/mnt/india
/root/Desktop/ /root/Desktop/
```

rsync (Remote Sync)

rsync is a most commonly used command for copying and synchronizing files and directories remotely as well as locally in Linux/Unix systems.

Transfer file local to remote server

```
#rsync -rvh myfile1.txt root@<ip_Address>:/mnt/
```

Transfer file remote server to local

```
#rsync -rvh root@<ip_Address>:/mnt/myfile1.txt
/root/Desktop/
```

. . .

NIC Teaming (Bonding) / Link Aggregation

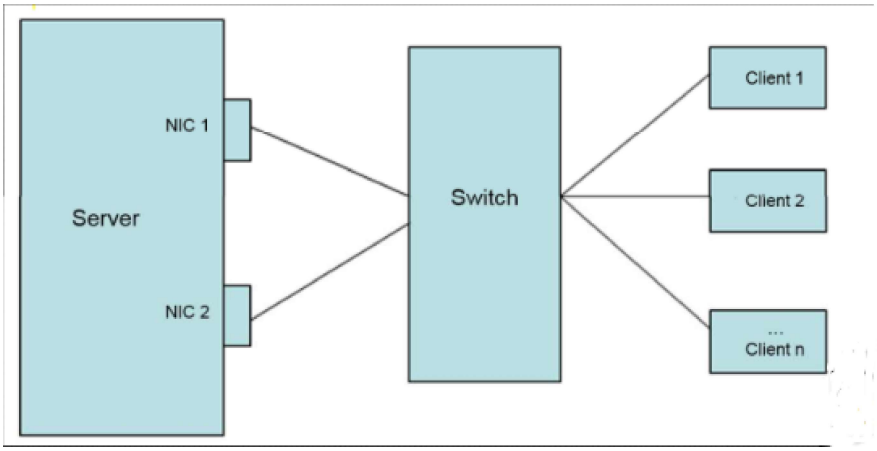
What is NIC Teaming

- Network interface bonding is called by many names: Port Trunking, Channel Bonding, Link Aggregation, NIC teaming, and others. It combines or aggregates multiple network connections into a single channel bonding interface. This allows two or more network interfaces to act as one, to increase throughput and to provide redundancy or failover.
- The Linux kernel comes with the bonding driver for aggregating multiple physical network interfaces into a single logical interface (for example, aggregating eth0 and eth1 into bond0).
- NIC teaming is the concept of combining or bonding 2 or more network
  interfaces into one logical interface to provide high throughput and
  redundancy. This practice is popular especially with critical

servers where
high availability is expected at all times.

- In a server with 2 or more NIC cards, the concept of NIC teaming is critical in the event where one NIC card fails. With NIC teaming, the logical network interface will ensure that the remaining NIC will continue functioning and serving the purpose of the defective NIC.

NIC Teaming



Available runners (Team Running modes)

Runners:
These are distinct units of implement NIC teaming in different mode.

broadcast—data is transmitted over all ports

active-backup—one port or link is used while others are kept as a backup

round-robin—data is transmitted over all ports in turn

loadbalance—Traffic is distributed across all NIC's.

Advantage of NIC Teaming

Load balancing
In the case of NIC teaming, the network traffic is balanced across all active NICs equally.

Fault tolerance

Another benefit offered by NIC teaming is higher fault tolerance. If one of the
underlying physical NICs is broken down or if the cable of the corresponding
NIC is unplugged, the host/server detects the fault condition and moves the traffic to another NIC automatically. This reduces the possibility of a breakdown of the entire network, thus improving the fault tolerance of the system.

Configure network teaming daemon

```
#yum install teamd
```

Create the taeaming interface:

```
#nmcli con add type team con-name Team1 ifname Team1 config
‘{“runner“: {“name”: “activebackup”}}’
```

Add an IPv4 Configuration

```
#nmcli con modify Team1 ipv4.add 192.168.1.10/24 gw4 192.168.1.1
ipv4.dns 192.168.1.1 connection.autoconnect yes ipv4.method manual
```

Add the eth0 & eth1 interface to the teaming interface

```
#nmcli con add type team-slave con-name Team1-slave1 ifname enp0s1
master Team1
```

```
#nmcli con add type team-slave con-name Team1-slave2 ifname enp0s2
master
Team1
```

Activate the teaming interface:

```
#nmcli con up Team1
#nmcli con up Team1-slave1
#nmcli con up Team1-slave2
```

For check:

```
#ping 192.168.1.10
#teamdctl Team1 stat
```

• • •