Automating Installation with kickstart

- A system administrator can automate the installation of Red Hat Enterprise Linux using a feature called kickstart. Anaconda, the Red Hat Installer, needs to be told how to install a systems; partition disks, configure network interfaces, select which packages to install, etc. A kickstart installation uses a text file to provide all of the answers to these questions, so no interaction is required.

- Kickstart in RHEL is similar to Jumpstart in Solaris.

- We can use Kickstart method to install N number of servers yet the same time because this method does not required user intervention while installation process

Automating Installation with kickstart

- We have to create a kickstart config file which contains all answers for operating system installation.

Automated OS Installation Advantages

- **We can install multiple servers in one go or simultaneously**
- **Minimize manual intervention and save time**
- **Multiple Distributions also supported**
- **Post installation scripts helps in automating more tasks**

Automating Installation with kickstart

- Kickstart Method uses below protocols for the installation of OS automatically

NFS
HTTP
FTP

➢Automating Installation with kickstart

Step 1: Mount ISO and dump Media Source file

Here, I am using ftp for kickstart method installation but we can also use http, https & nfs as well.

Mounting the CD Drive

 # mount /dev/sr0 /mnt

FTP server Path: cp -rfv /mnt/* /var/ftp/pub/

➢Automating Installation with kickstart

Step 2: Installation and generate Kickstart file

First we need to install the package "system-config-kickstart" which can be used to configure the kickstart file.

 # yum install system-config-kickstart

Automating Installation with kickstart

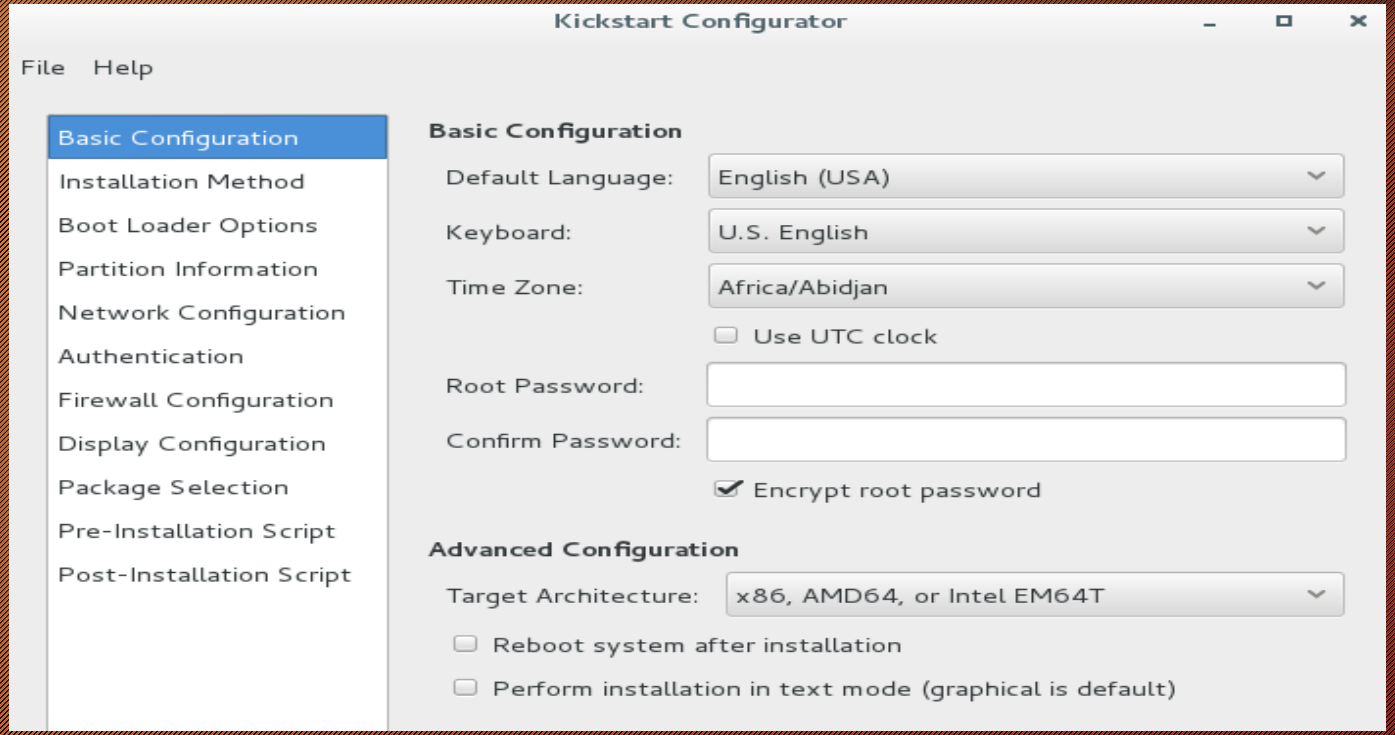
```
[root@Server1 ~]# yum install system-config-kickstart
Loaded plugins: langpacks, product-id, search-disabled-repos, subscription-
              : manager
This system is not registered to Red Hat Subscription Management. You can use subscription-manager to register.
Resolving Dependencies
There are unfinished transactions remaining. You might consider running yum-complete-transaction, or "yum-complete-transaction --
cleanup-only" and "yum history redo last", first to finish them. If those don't work you'll have to try removing/installing packages
by hand (maybe package-cleanup can help).
--> Running transaction check
---> Package system-config-kickstart.noarch 0:2.9.2-4.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                    Arch      Version           Repository         Size
================================================================================
Installing:
 system-config-kickstart    noarch    2.9.2-4.el7       RHEL7YumServer    351 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 351 k
Installed size: 2.0 M
Is this ok [y/d/N]:
```

Step 3: Generating kick start conf file

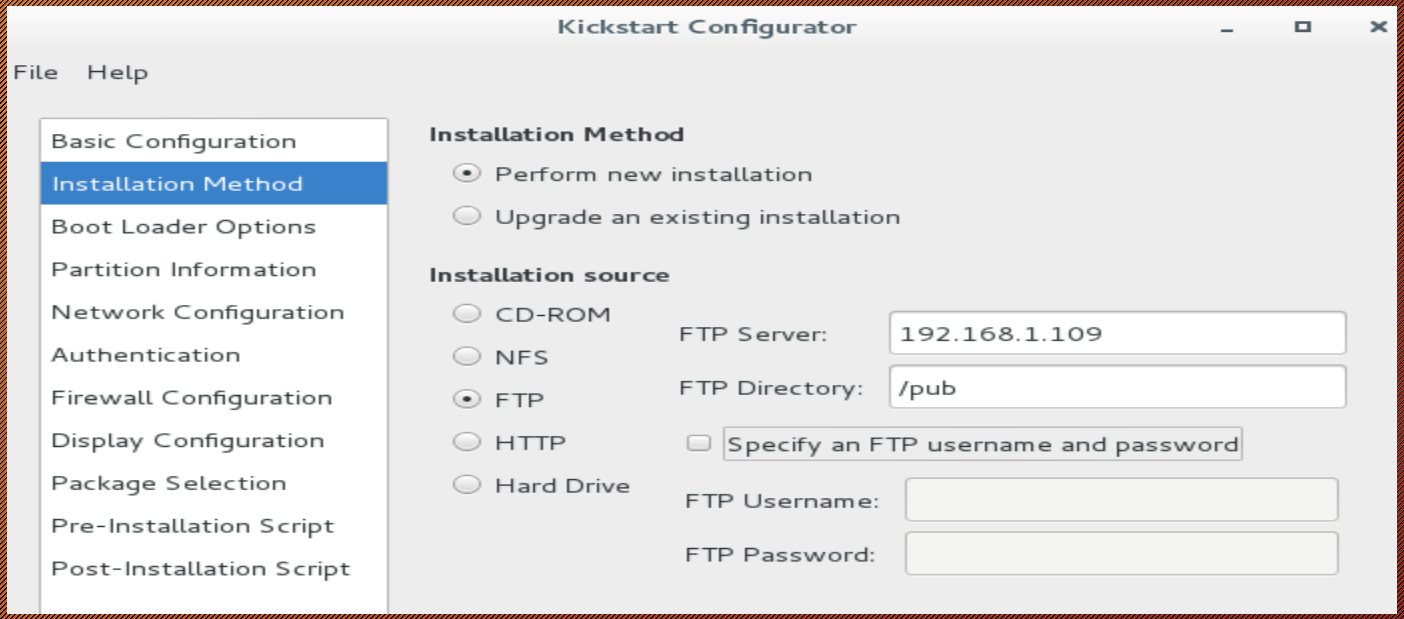
```
[root@Server1 ~]# system-config-kickstart
```

As shown in above screenshot select required options

- Default Language
- Keyboard
- Time Zone
- Root Password and Confirm Password
- Target Architecture
- Reboot system After Installation


Kickstart Configurator
File
Help
Basic Configuration
Installation Method
Boot Loader Options
Partition Information
Network Configuration
Authentication
Firewall Configuration
Display Configuration
Package Selection
Pre-Installation Script
Post-Installation Script
Basic Configuration
Default Language:
English (USA)
Keyboard:
U.S. English
Time Zone:
Africa/Abidjan
Use UTC clock
Root Password:
Confirm Password:
Encrypt root password
Advanced Configuration
Target Architecture:
x86, AMD64, or Intel EM64T
Reboot system after installation
Perform installation in text mode (graphical is default)

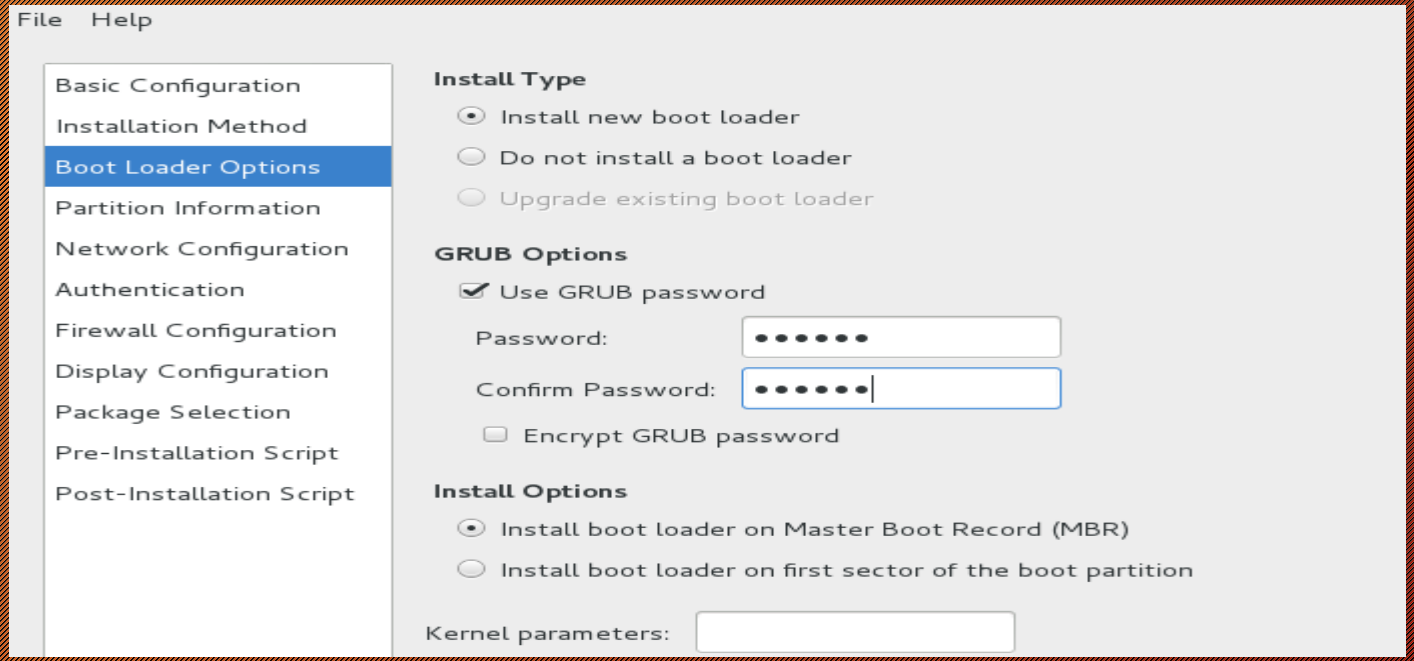
- **Step 3:** Select Installation method either perform fresh installation or upgrade an existing installation. Kickstart supports upgrade option as well.

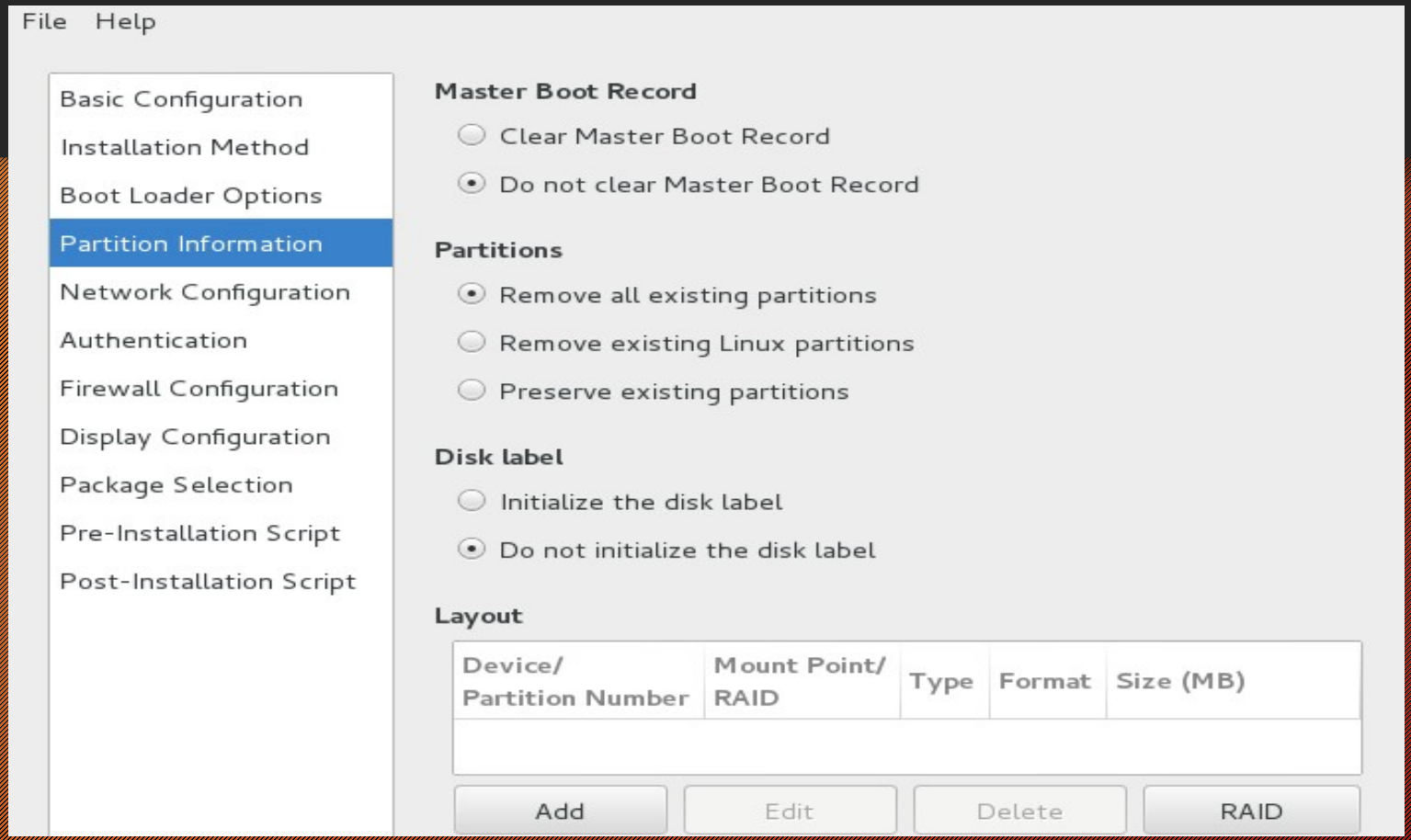


Step 3: Generating kick start conf file

If you would like to install new boot loader then select to install new boot loader or else select do not install boot loader. If your interested in setting up the GRUB password you can also do that by selecting the GRUB password option and provide password.

Step 3: Generating kick start conf file




File Help
Basic Configuration
Installation Method
Boot Loader Options
Partition Information
Network Configuration
Authentication
Firewall Configuration
Display Configuration
Package Selection
Pre-Installation Script
Post-Installation Script
Master Boot Record
Clear Master Boot Record
Do not clear Master Boot Record
Partitions
Remove all existing partitions
Remove existing Linux partitions
Preserve existing partitions
Disk label
Initialize the disk label
Do not initialize the disk label
Layout
Device/ Partition Number
Mount Point/ RAID
Type
Format
Size (MB)
Add
Edit
Delete
RAID

Step 3: Generating kick start conf file

Using Partition Information tab declare partition details which are the partitions you would like to create.

Clear Master Boot Record — Will clear before boot record if any

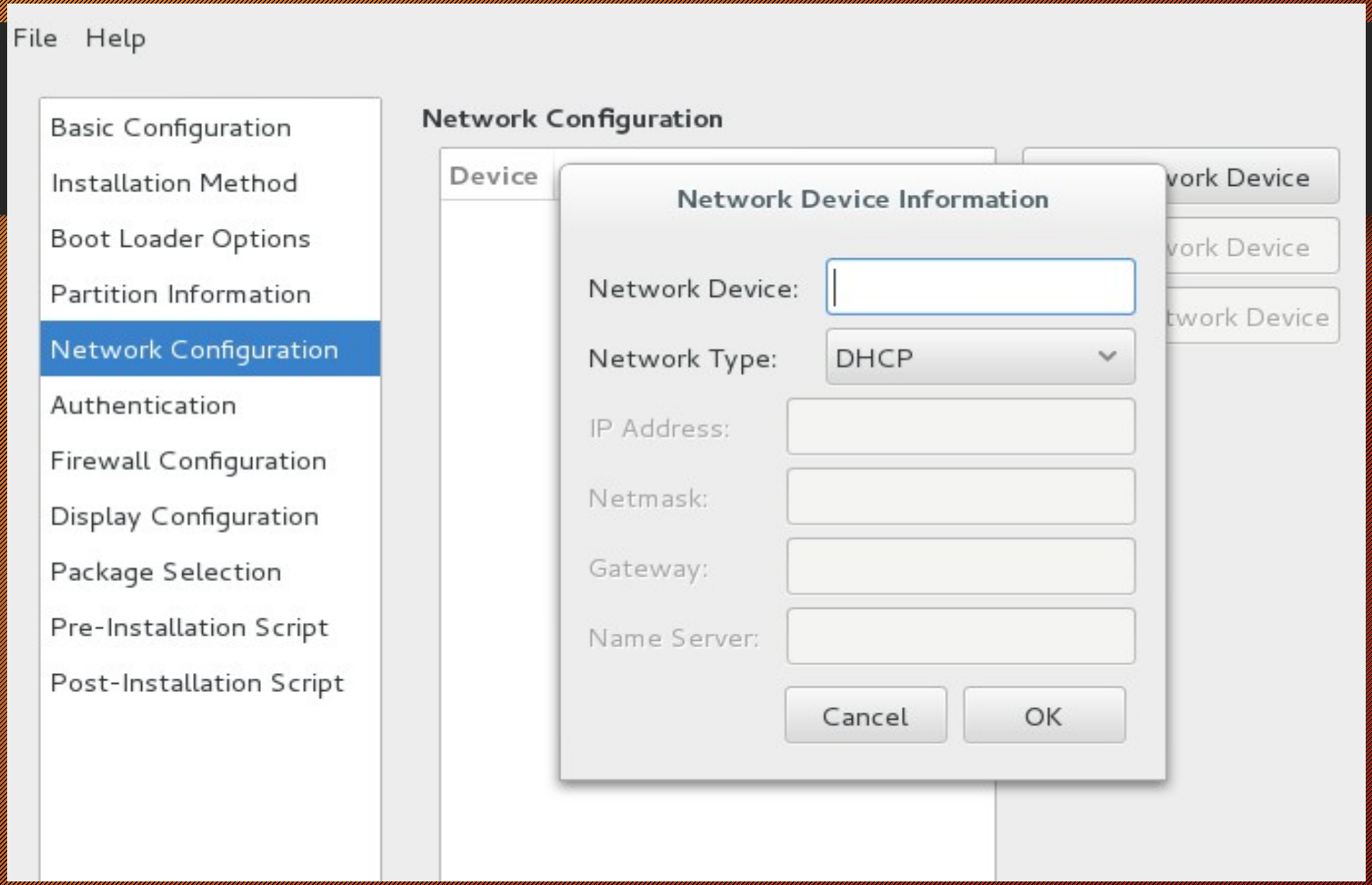
Do Not Clear Master Boot Record — It will not touch the previous boot record

Remove all – Will remove all existing partitions and create new

Remove only existing Linux partitions – it will not delete NTFS partitions

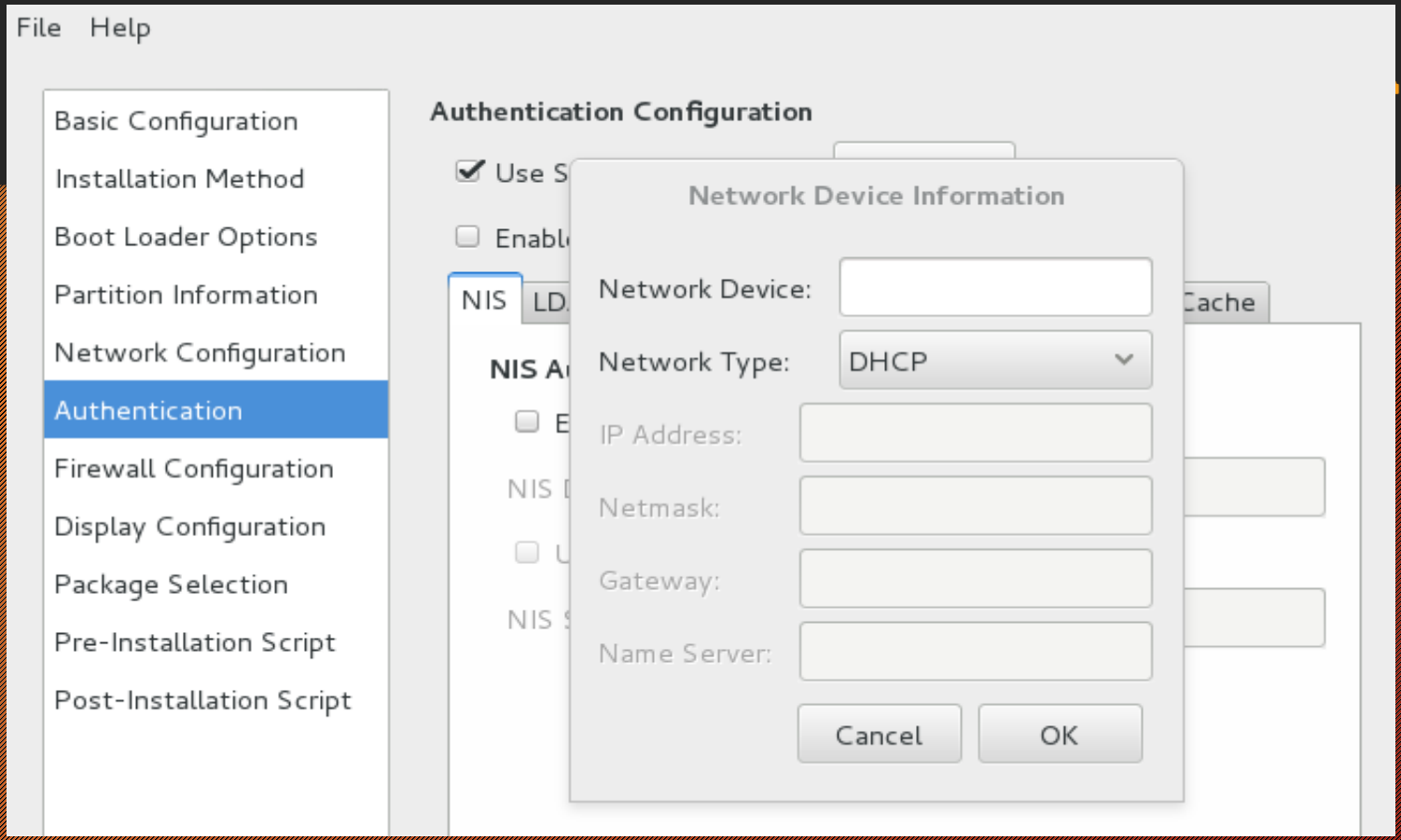
Preserve existing partitions – Will not touch any of existing partitions

Note: There is no option to create LVM partitions in this tool, Add config definition after file generation.


File Help
Basic Configuration
Installation Method
Boot Loader Options
Partition Information
Network Configuration
Authentication
Firewall Configuration
Display Configuration
Package Selection
Pre-Installation Script
Post-Installation Script
Network Configuration
Device
Network Device Information
Network Device:
Network Type:
DHCP
IP Address:
Netmask:
Gateway:
Name Server:
Cancel
OK

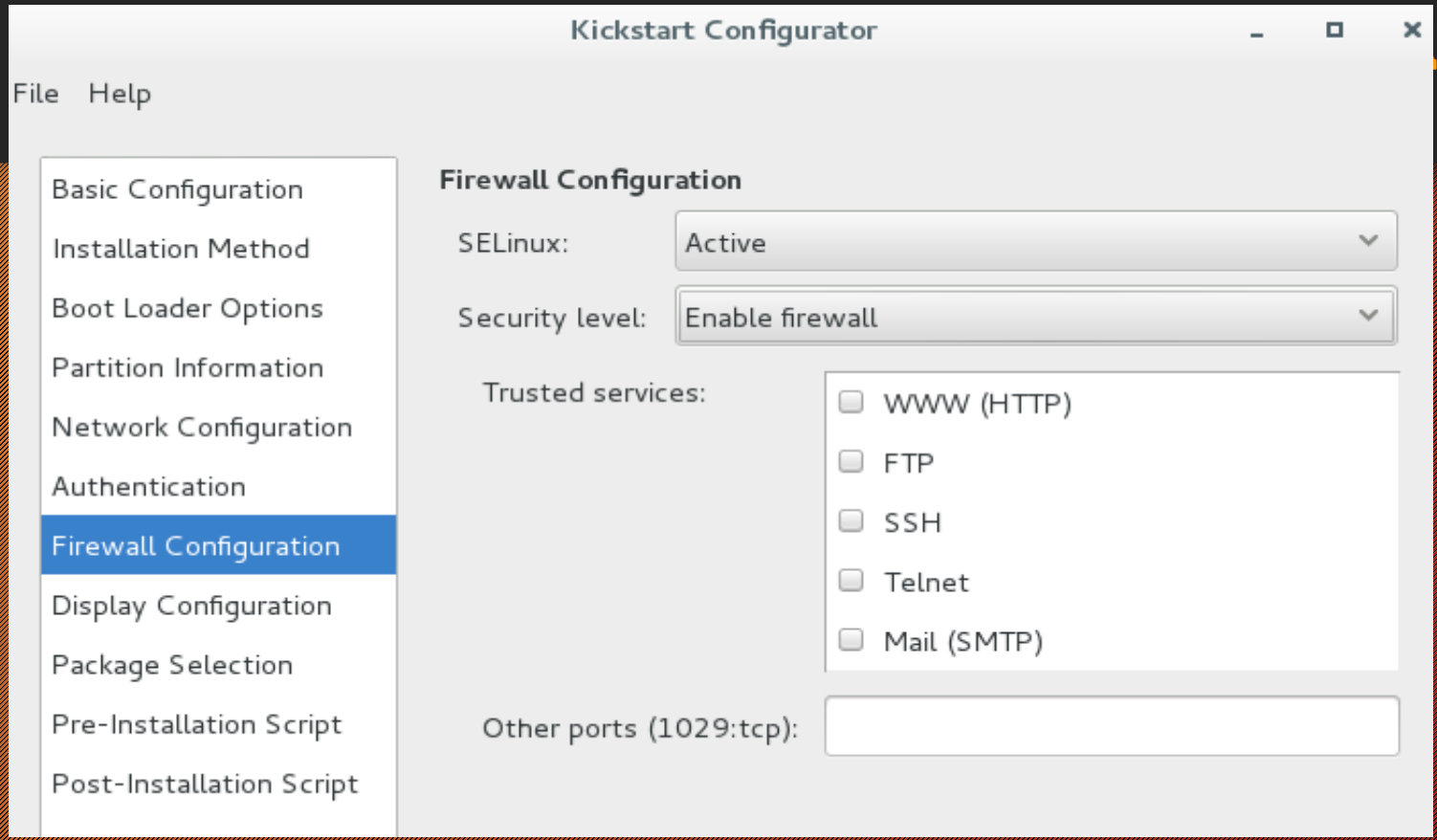
Step 3: Generating kick start conf file

Authentication Configuration is the option where you can select the option to join to NIS, LDAP, Kerberos and local encrypted authentication


File Help
Basic Configuration
Installation Method
Boot Loader Options
Partition Information
Network Configuration
Authentication
Firewall Configuration
Display Configuration
Package Selection
Pre-Installation Script
Post-Installation Script
Authentication Configuration
Use S
Enabl
NIS
LD
Cache
NIS A
E
NIS D
U
NIS S
Network Device Information
Network Device:
Network Type:
DHCP
IP Address:
Netmask:
Gateway:
Name Server:
Cancel
OK

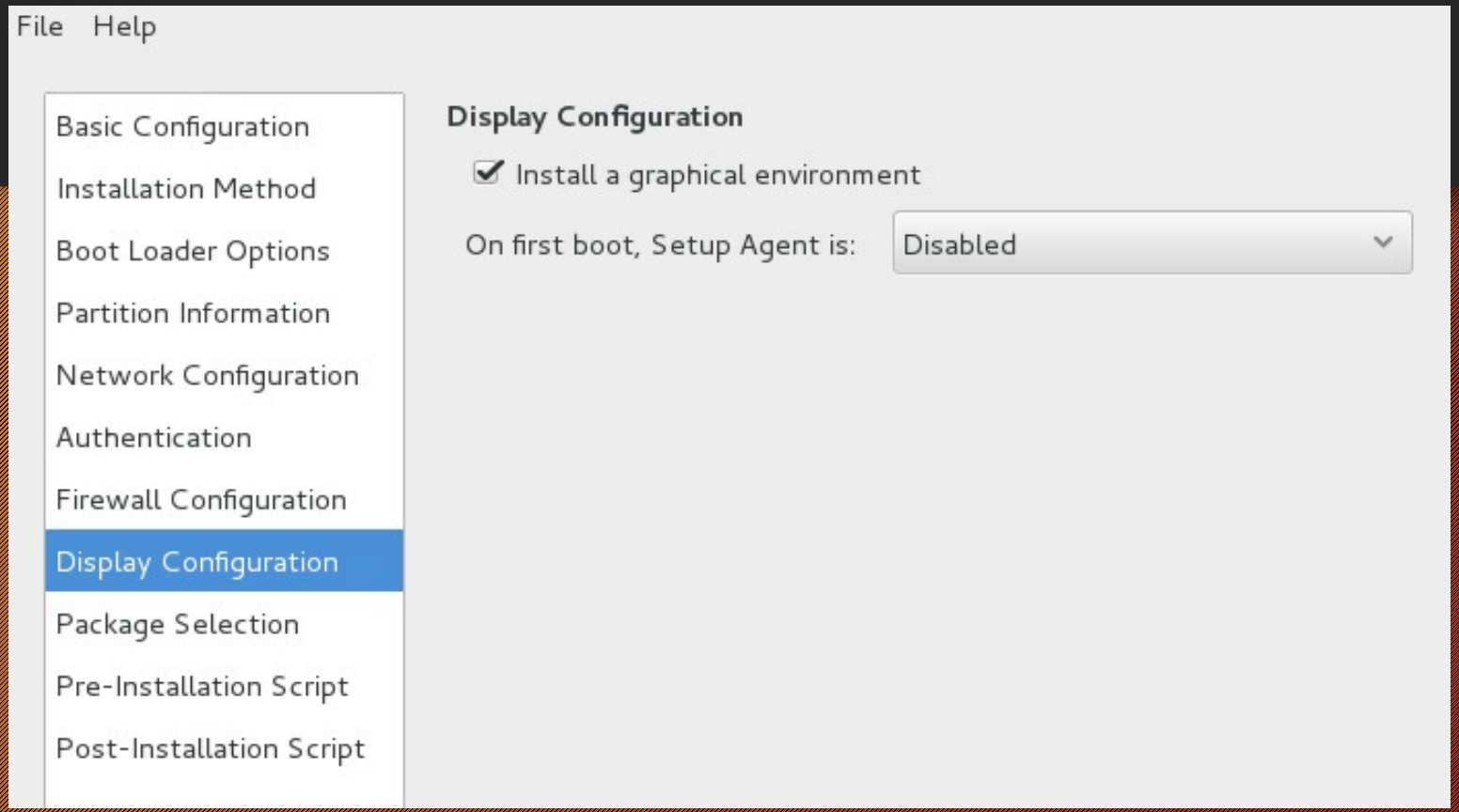
Step 3: Generating kick start conf file

- Decision to Enable / Disable firewall and its security level


Kickstart Configurator
File
Help
Basic Configuration
Installation Method
Boot Loader Options
Partition Information
Network Configuration
Authentication
Firewall Configuration
Display Configuration
Package Selection
Pre-Installation Script
Post-Installation Script
Firewall Configuration
SELinux:
Active
Security level:
Enable firewall
Trusted services:
WWW (HTTP)
FTP
SSH
Telnet
Mail (SMTP)
Other ports (1029:tcp):

Step 3: Generating kick start conf file

If you would like to install graphical environment then simply select option


File Help
Basic Configuration
Installation Method
Boot Loader Options
Partition Information
Network Configuration
Authentication
Firewall Configuration
Display Configuration
Package Selection
Pre-Installation Script
Post-Installation Script
Display Configuration
Install a graphical environment
On first boot, Setup Agent is:
Disabled

Step 3: Generating kick start conf file

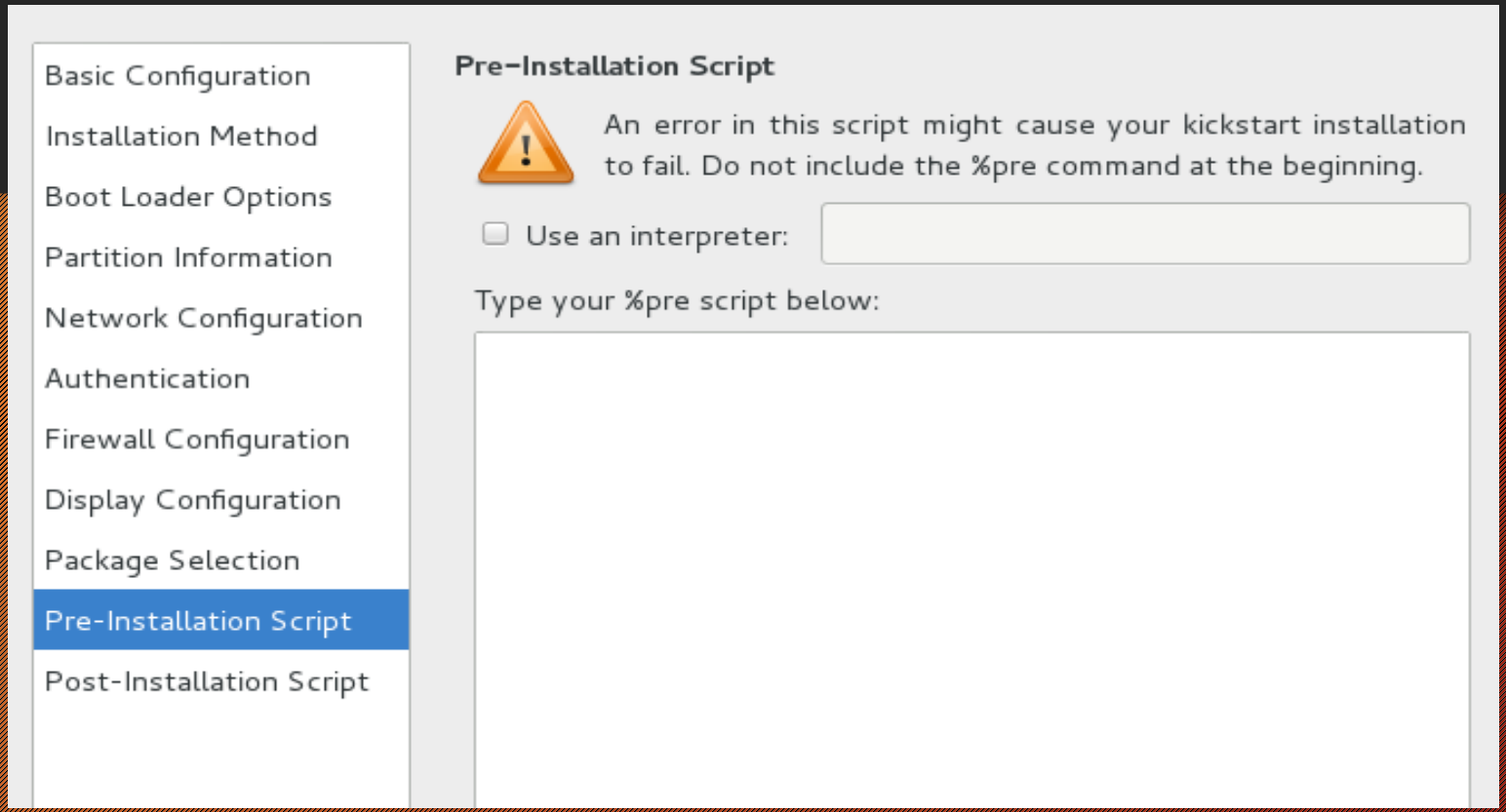
Note: Package option did not have an option to select packages, after generating config file will add

Step 3: Generating kick start conf file

Note: Package option did not have an option to select packages, after generating config file will add

Pre-Installation Script

Post-Installation Script


Basic Configuration
Installation Method
Boot Loader Options
Partition Information
Network Configuration
Authentication
Firewall Configuration
Display Configuration
Package Selection
Pre-Installation Script
Post-Installation Script
Pre-Installation Script
An error in this script might cause your kickstart installation to fail. Do not include the %pre command at the beginning.
Use an interpreter:
Type your %pre script below:

Step4: Adding LVM config and Packages list

Simply generating Kickstart file will not work as expected, we have to add LVM configuration and Packages which you would like to install, edit config file and add

Edit config file and add below lines to create LVM partitions and packages installation Automated OS

```
# Partition clearing information
clearpart --all --initlabel
volgroup rhel --pesize=4096 PV0
part PV0 --fstype=lvmpv --ondisk=sda --size=50000
part /boot --fstype=xfs --size=500
logvol / --vgname=rhel --name=root --fstype=xfs --size=10000
logvol /var --vgname=rhel --name=var --fstype=xfs --size=8000
logvol swap --vgname=rhel --name=swap --fstype=swap --size=8000
logvol /home --vgname=rhel --name=home --fstype=xfs --size=7000
logvol /usr --vgname=rhel --name=usr --fstype=xfs --size=7000
%packages
@base
@core
@desktop-debugging
@dial-up
@fonts
@gnome-desktop
@guest-agents
@guest-desktop-agents
@input-methods
@internet-browser
@mariadb
@multimedia
@print-client
@x11
kexec-tools
```

❑ Steps for FTP Protocol Linux RHEL7

➢ Installing required packages for FTP and start ftp service

```
# yum install vsftpd*
#systemctl enable vsftpd.service
ln -s '/usr/lib/systemd/system/vsftpd.service' '/etc/systemd/system/multi-user.target.wants/vsftpd.service'
#systemctl start vsftpd.service
#systemctl status vsftpd.service
vsftpd.service - Vsftpd ftp daemon
Loaded: loaded (/usr/lib/systemd/system/vsftpd.service; enabled)
Active: active (running) since Sat 2017-02-25 14:57:42 IST; 7s ago
Process: 12135 ExecStart=/usr/sbin/vsftpd /etc/vsftpd/vsftpd.conf (code=exited, status=0/SUCCESS)
Main PID: 10795 (vsftpd)
```

- **Steps for FTP Protocol Linux RHEL7**

  - **Installing required packages for FTP and start ftp service**

#firewall-cmd --permanent --add-service=ftp
success
#firewall-cmd --reload
success

- Copy Kickstart file and Media files to /var/ftp/pub/

- **Client Side OS Installation**

- **Boot client machine with OS CD/DVD**

  - **boot: linux ks=http://192.168.0.109/pub/ks.cfg**

Network Teaming

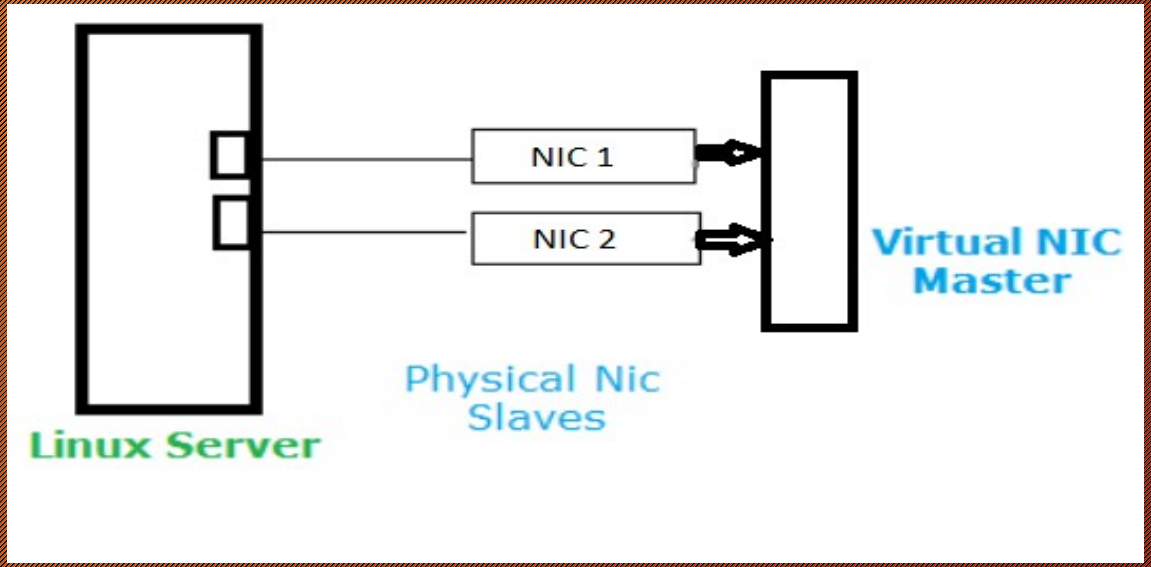
- The Linux NIC teaming driver provides a method for aggregating multiple network interfaces into a single “Team” interface.

- Network teaming is method for linking NICs together logically to allow for failover or higher throughout.

- RHEL 7 supports channel bonding for backward compatibility. Network teaming provides better performance and is more extensible because of its modular design.

- The behavior of the team interfaces depends upon the runner configuration. Generally teaming supports below methods.

Network Teaming



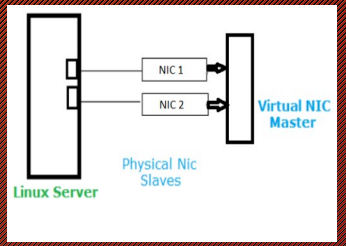
Network Teaming

- NIC Teaming supports
  - Round robin NIC Teaming
  - Load balancing NIC Teaming
  - Fail-over NIC Teaming
  - Broadcast NIC Teaming

Network Teaming

- NIC Teaming supports

Mode	Policy	How it works	Fault Tolerance	Load balancing
1	Round Robin	packets are sequentially transmitted/received through each interfaces one by one.	No	Yes
2	Active Backup	one NIC active while another NIC is asleep. If the active NIC goes down, another NIC becomes active. only supported in x86 environments.	Yes	No



Network Teaming

- NIC Teaming supports

Mode	Policy	How it works	Fault Tolerance	Load balancing
3	Broadcast	All transmissions are sent on all slaves	Yes	No
4	Loadbalancer	This runner monitors traffic and uses a hash function to try to reach a perfect balance when selecting ports for packet transmission.	Yes	Yes

Network Teaming

- NIC Teaming supports

Mode	Policy	How it works	Fault Tolerance	Load balancing
5	Dynamic Link Aggregation (lacp)	aggregated NICs act as one NIC which results in a higher throughput, but also provides failover in the case that a NIC fails. Dynamic Link Aggregation requires a switch that supports IEEE 802.3ad.	Yes	Yes

Network Teaming

- **NIC Teaming supports**

- The key reasons why you might want to use teaming rather than bonding are

  - Teaming has a small kernel module which implements fast handling of packets flowing through your teamed interfaces
  - support for IPv6 (NS/NA) link monitoring
  - Capable of working with D-Bus and Unix Domain Sockets (the default)
  - It provides an extensible and scale-able solution for your teaming requirements
  - load balancing for LACP support
  - It makes use of NetworkManager and its associates tools (the modern way) to manage your your network interfaces

Network Teaming

- ✓Practice Lab Session

Network Teaming

- Practice Lab Session

Creating teaming virtual interface – (how to configure NIC Teaming as Active Backup (Failover)

Configuring iSCSI Target & Initiator

- Internet Small Computer System Interface (iSCSI) is an TCP/IP-based standard for connecting storage devices. iSCSI uses IP networks to encapsulate SCSI commands, allowing data to be transferred over long distances.

- iSCSI provides shared storage among a number of client systems. Storage devices are attached to servers (targets). Client systems (initiators) access the remote storage devices over IP networks.

- To the client systems, the storage devices appear to be locally attached. iSCSI uses the existing IP infrastructure and does not require any additional cabling, as is the case with Fibre Channel (FC) storage area networks.

Configuring iSCSI Target & Initiator

iSCSI fundamentals

iSCSI Component Terminology

Term	Description
initiator	An iSCSI client, typically available as software but also implemented as iSCSI HBAs. Initiators must be given unique names (see **IQN**).
target	An iSCSI storage resource, configured for connection from an iSCSI server. Targets must be given unique names (see **IQN**). A target provides one or more numbered block devices called logical units (see **LUN**). An iSCSI server can provide many targets concurrently.
ACL	An Access Control List (entry), an access restriction using the node IQN (commonly the iSCSI Initiator Name) to validate access permissions for an initiator.
discovery	Querying a target server to list configured targets. Target use requires an additional access steps (see **login**).

Configuring iSCSI Target & Initiator

iSCSI fundamentals

IQN	An iSCSI Qualified Name, a worldwide unique name used to identify both initiators and targets, in the mandated naming format: **iqn.YYYY-MM.com.reversed.domain[:optional_string]** **iqn**—Signifying that this name will use a domain as its identifier. **YYYY-MM**—The first month in which the domain name was owned. **com.reversed.domain**—The reversed domain name of the organization creating this iSCSI name. **optional_string**—An optional, colon-prefixed string assigned by the domain owner as desired while remaining worldwide unique. It may include colons to separate organization boundaries.
login	Authenticating to a target or LUN to begin client block device use.
LUN	A Logical Unit Number, numbered block devices attached to and available through a target. One or more LUNs may be attached to a single target, although typically a target provides only one LUN.
node	Any iSCSI initiator or iSCSI target, identified by its IQN.
portal	An IP address and port on a target or initiator used to establish connections. Some iSCSI implementations use *portal* and *node* interchangeably.
TPG	Target Portal Group, the set of interface IP addresses and TCP ports to which a specific iSCSI target will listen. Target configuration (e.g., ACLs) can be added to the TPG to coordinate settings for multiple LUNs.

Configuring iSCSI Target & Initiator

Configuring an iSCSI Server

- RHEL/CentOS 7 uses the Linux-IO (LIO) kernel target subsystem for iSCSI. In addition to iSCSI, LIO supports a number of storage fabrics including Fibre Channel over Ethernet (FCoE).
- In RHEL 7, all storage fabrics are managed with the **targetcli** utility.

- To configure RHEL system as an iSCSI server, begin by installing the targetcli software package:
- yum install targetcli

- Installing the targetcli software package also installs the python-rtslib package, which provides the /usr/lib/systemd/system/target.service file.

Configuring iSCSI Target & Initiator

Configuring an iSCSI Server

➢Before using the targetcli utility to create, delete, and view storage targets, use the systemctl command to enable and start the target service on the iSCSI server.

- systemctl start target
- systemctl status target
- systemctl enable target

Configuring iSCSI Target & Initiator

Configuring an iSCSI Server

- **targetcli Utility**

- The targetcli utility is the administration shell for creating, editing, and viewing the configuration of the kernel's target subsystem. Run targetcli to enter the configuration shell.

```
targetcli
Warning: Could not load preferences file /root/.targetcli/prefs.bin.
targetcli shell version 2.2.fb56
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.
/> help
```

Configuring iSCSI Target & Initiator

Configuring an iSCSI Server

- **Backstores**

- The first thing listed are the backstores. Backstores provide different ways of storing the data locally that will be exported to an external system. The available options are block, fileio, pscsi and ramdisk.

- The mappings to local storage resources that each backstore creates are called storage objects. Use the targetcli ls command to list the different types of backstores.

Create backing storage (backstores). There are several types of backing storage.

- **`block`** - A block device defined on the server. A disk drive, disk partition, a logical volume, multipath device, any device files defined on the server that are of type **b**.
- **`fileio`** - Create a file, of a specified size, in the filesystem of the server. This method is similar to using image files to be the storage for virtual machine disk images.
- **`pscsi`** - Physical SCSI. Permits passthrough to a physical SCSI device connected to the server. This backstore type is not typically used.
- **`ramdisk`** - Create a ramdisk device, of a specified size, in memory on the server. This type of storage will not store data persistently. When the server is rebooted, the ramdisk definition will return when the target is instantiated, but all data will have been lost.

Configuring iSCSI Target & Initiator

- Configuring an iSCSI Server

- To create a block backstore from the targetcli shell:

```
/> cd /backstores/block
/backstores/block> create <name of shared lun> /dev/xxx
/backstores/block> create sharedlun0 /dev/sde
```

To create a fileio backstore from the targetcli shell:

```
/> cd /backstores/fileio
/backstores/fileio> create name=LUN_3 /root/disk1.img 5G
```

Configuring iSCSI Target & Initiator

- Creating an iSCSI Target

- To create an iSCSI target from the targetcli shell, use the cd command to change to the /iscsi directory.

- /> cd /iscsi
- /iscsi>

Configuring iSCSI Target & Initiator

- **Creating an iSCSI Target**

- Use the create command without any arguments to create an iSCSI target by using a default target name. By default, the target is identified by an “iqn” identifier. This is an iSCSI Qualified Name (IQN), which uniquely identifies a target. IQN format addresses are most commonly used to identify a target. This address consists of the following fields:

  Literal iqn

  Date (in yyyy-mm format) that the naming authority took ownership of the domain

  Reversed domain name of the authority

  Optional “:” that prefixes a storage target name specified by the naming authority

Configuring iSCSI Target & Initiator

➢Creating an iSCSI Target

```
/> cd /iscsi
/iscsi> create
Created target iqn.2007-02.org.linux-iscsi.user.x8664:sn.b0dd6e768beb.
Created TPG 1.
Global pref auto_add_default_portal=true
Created default portal listening on all IPs (0.0.0.0), port 3260.
/iscsi>
```

Configuring iSCSI Target & Initiator

- To allow remote systems to access an iSCSI target on port 3260, either disable the firewalld service on the iSCSI server or configure firewalld to trust the 3260/tcp port. The following example uses firewall-cmd to open the 3260/tcp port for the firewalld service.

- # firewall-cmd --permanent --add-port=3260/tcp

- If you include the –permanent option when adding a port, use the firewall-cmd command to reload the configuration.

- # firewall-cmd –reload

Configuring iSCSI Target & Initiator

- Creating iSCSI LUNs

- The kernel target exports SCSI Logical Units to remote systems. Use the targetcli shell to link previously defined storage objects with a target, and to specify which Logical Unit Number (LUN) the device uses.

- /iscsi> cd /iscsi/iqn.2007-02.org.linux-iscsi.user.x8664:sn.b0dd6e768beb /
- /iscsi/iqn.20....b0dd6e768beb> cd tpgl/luns

- The following commands create a LUN from the previously defined block storage objects.

- /iscsi/iqn.20....b0dd6e768beb> create /backstores/block/sharedlun0
- Created LUN 1.

Configuring iSCSI Target & Initiator

- Creating ACLs

- Access Control Lists (ACLs) restrict access to LUNs from remote systems. You can create an ACL for each initiator to enforce authentication when the initiator connects to the target. This allows you to give a specific initiator exclusive access to a specific target. The following example uses the create command to create an ACL for an initiator. From the targetcli shell, begin by using the cd command to change to the acls directory within the [target/TGP] hierarchy.

- /> cd /iscsi/iqn.2007-02.org.linux-iscsi.user.x8664:sn.b0dd6e768beb/tpg1/acls
- /iscsi/iqn.20...beb/tpg1/acls> create iqn.2008-05.com.abcded:aabb51a64012 <= IQN of the client
- Created Node ACL for iqn.2005-05.com.abcded:aabb51a64012
- Created mapped LUN 1.

Configuring iSCSI Target & Initiator

- Configuring an iSCSI Initiator

- To configure a Linux system as an iSCSI initiator, install the iscsi-initiator-utils software package. This package is the Linux Open-iSCSI Initiator.
- yum install iscsi-initiator-utils

- The package installs several files including the following:

/etc/iscsi/iscsid.conf: The configuration file read by iscsid and iscsiadm. This file is heavily commented with descriptions for each configuration directive.

/sbin/iscsid: The Open-iSCSI daemon that implements the control path and management facilities

/sbin/iscsiadm: The Open-iSCSI administration utility used to discover and log in to iSCSI targets

❖Using Regular Expressions with Grep

After completing this selection, students should be able to

- Create regular expressions that match desired data.
- Use grep to apply regular expressions to text files