Red Hat Certified Engineer (RHCE)

RHCE 7 Summary

Repositories and Host Allowance/Denial

man 5 hosts_access

# vim /etc/yum.repos.d/rhce.repo

```
name=RHCE_RHEL7
baseurl=http://<baseurl>
enabled=1
gpgcheck=0
```

Can be done automatically with the command:

sudo yum-config-manager --add-repo http://server.example.com/rep cat /etc/yum.repos.d/server.example.com_repo.rep

```
[server.example.com_repo]
name=added from: http://server.example.com/repo
baseurl=http://server.example.com/repo
enabled=1
```

# yum repolist

1. Allow SSH for a domain and deny SSH to all the others:

# vim /etc/hosts.allow

```
sshd: .domain.com
```

# vim /etc/hosts.deny

```
sshd: ALL
```

2. Allow SSH for only specific IP and block all the others:

# vim /etc/hosts.deny

```
sshd: ALL EXCEPT 192.168.0.1
```

3. Denies all services to all hosts unless permitted in hosts.allow:

# vim /etc/hosts.allow

```
ALL: .foobar.edu EXCEPT terminalserver.foobar.edu
```

# vim /etc/hosts.deny

```
ALL
```

4. Access granted by default, redundant file hosts.allow

# vim /etc/hosts.deny

```
some.host.name, .some.domain
```

# vim /etc/hosts.deny

```
ALL EXCEPT in.fingerd: other.host.name, .other.domain
```

5. Rules can be also only in one file, for example:

# vim /etc/hosts.allow

```
ALL: .friendly.domain: ALLOW
ALL: ALL: DENY
```

# vim /etc/hosts.allow

```
ALL: .bad.domain: DENY
ALL: ALL: ALLOW
```

Recover root password

```
reboot
e
linux16...
rd.break enforcing=0
ctrl+x
switch_root:/# mount -oremount,rw /sysroot
switch_root:/# chroot /sysroot
sh-4.2# passwd root
Changing password for user root.
New passwd: mypassword
Retype new password: mypassword
passwd: all authentication token updated
successfully.
sh-4.2# exit
switch_root:/# exit
logout

...
[  OK  ] Started Network Manager Script Dispatcher
Service.
[  OK  ] Started Crash recovery kernel arming.
[  OK  ] Reached target Multi-User System.

CentOS Linux 7 (Core)
Kernel 3.10.0-229.14.1.el7.x86_64 on an x86_64

vm login: root
Password: mypassword

# restorecon /etc/shadow
# setenforce enforcing
```

SERVICES

```
systemctl --failed --type=service
systemctl show <unit>
systemctl status <-l> <unit> <-l>
systemctl stop|start|restart|reload <unit>
systemctl mask|unmask <unit>
systemctl enable|disable <unit>
systemctl list-dependencies <unit>
systemctl list-units --type=service --all
systemctl list-unit-files --type=service
systemctl get-default
systemctl set-default <graphical|multi-
user|rescue|emergency>
systemctl isolate <graphical|multi-
user|rescue|emergency>
```

IPV4

```
nmcli dev status
nmcli con show <name>
nmcli con show --active
ip addr show <eth0> / ip a
ip link / ip l
nmcli con add con-name <name> type ethernet ifname
<eth0> ip4 xxx.xxx.xxx.xxx/24 gw4 xxx.xxx.xxx.xxx
nmcli con mod <name> ipv4.addresses "192.0.2.2/24
192.0.2.254"
nmcli con <up|down> <name>
nmcli dev status
nmcli dev dis <eth0>
nmcli con mod <name> +ipv4.dns xxx.xxx.xxx.xxx
        vim /etc/sysconfig/network-script/ifcfg-
<name>
nmcli con reload
nmcli con del <name>
```

```
hostname
hostnamectl set-hostname <name>
        vim /etc/hostname
hostnamectl status
ip route / ip r
ss -tulpn | grep sshd
```

IPV6

```
nmcli con add con-name <name> type ethernet ifname
<eth0> ip6 xxxx:xxxx:xxx:x:x:x/64 gw6
xxxx:xxxx:xxx:x:x:x
ip -6 route show
ping6 xxxx:xxxx:xxx:x:x:x
ping6 xxxx:xxxx:xxx:x:x:x<%eth1> for link-local
addresses and multicast groups
tracepath6 xxxx:xxxx:xxx:x:x:x
ss -A inet -n
netstat -46n
nmcli con mod <name> ipv6.method manual
```

TEAMING

man 5 nmcli-examples man 5 teamd.conf /usr/share/doc/teamd-1.25

```
nmcli con add con-name <team0> type team ifname
<team0> config '{ "runner": { "name": "
<activebackup|broadcast|loadbalance|roundrobin|lacp>"
}}'
```

Must be before ipv4.method

```
nmcli con mod <team0> ipv4.address xxx.xxx.xx.x/24
```

```
nmcli con mod <team0> ipv4.method manual
nmcli con mod <team0> connection.autoconnect yes
```

or autoconect yes during con add

```
nmcli con add con-name <team0-port1> type team-slave
ifname <eth0> master <team0>
nmcli con add con-name <team0-port2> type team-slave
ifname <eth1> master <team0>
```

-con-name <teamX-portX> not necessary, default is team-slave-<IFACE>

```
nmcli con up <team0>
nmcli con up team0-port1
nmcli con up team0-port2
nmcli dev dis eth1
teamdctl <team0> state
teamdctl <team0> config dump
teamnl <team0> ports
teamnl <team0> options
teamnl <team0> getoption activeport
teamnl <team0> setoption activeport <2>
```

If you make a mistake:

```
nmcli con mod <team0> team.config '{"runner":
{"name":"activebackup"}}'
```

BRIDGING

```
nmcli con add con-name <bridge0> type bridge ifname
<br0>
b/ nmcli con add con-name <bridge0-port1> type
bridge-slave ifname <eth0> master <br0>
c/ nmcli con add con-name <bridge0-port2> type
bridge-slave ifname <eth1> master <br0>
brctl show
```

BRIDGE=brteam0 /etc/sysconfig/network-scripts/ifcfg-team

FIREWALL

man 5 firewalld.richlanguage

Understand Zones

man firewalld.zones

```
systemctl mask <iptables|ip6tables|ebtables>
firewall-cmd --set-default zone=
<dmz|trusted|home|internal|work|public|external|block
|drop>
```

- **trusted**=all incoming traffic allowed

- **home**=reject incoming unless matching outgoing, accept incoming ssh,mdns,ipp-client,samba-client,dhcpv6-client
- **internal**=same as home
- **work**=reject incoming unless matching outgoing, accept incoming ssh,ipp-client,dhcpv6-client
- **public**=reject incoming unless matching outgoing, accept incoming ssh,dhcpv6-client *[DEFAULT]*
- **external**=reject incoming unless matching outgoing, accept incoming ssh, masquerading enabled
- **dmz**=reject incoming unless matching outgoing, accept incoming ssh
- **block**=reject incoming unless matching outgoing
- **drop**=reject incoming unless matching outgoing, does not respond at all

Rules

/etc/firewall.d; /usr/lib/firewalld

```
firewall-cmd --<get-default-zone|set-default-
zone|get-zones|get-services|get-active-zones|list-
all>
firewall-cmd --<add|remove-rich-rule=RULE|query-rich-
rule=RULE|list-rich-rules>
firewall-cmd --<remove-service=SERVICE|remove-
port=PORT/PROTOCOL>
firewall-cmd --permanent --zone=<name> --add-
source=xxx.xxx.xx.x/24
firewall-cmd --timeout=60 --zone=<name> --add-
service=mysql
firewall-cmd --reload
firewall-cmd --remove-service=haproxy –zone=public
firewall-cmd --direct --get-all-rules
firewall-cmd --get-zone-of-interface=eth0
```

Rich Rules rule source destination [service|port|masquerade|forward-port] log audit

```
firewall-cmd --permanent --zone=<name> --add-rich-
rule='rule family=ipv4 source address=xxx.xxx.xx.x/32
reject'
firewall-cmd --permanent --zone=<name> --add-rich-
rule='rule family=ipv4 source address=xxx.xxx.xx.x/24
port=xxxx-xxxx protocol tcp <accept|reject|drop>'
firewall-cmd --add-rich-rule='rule service name=ftp
limit value=2/m accept'
firewall-cmd --permanent --zone=<name> --add-
masquerade
firewall-cmd --permanent --zone=<name> --add-rich-
rule='rule family=ipv4 source address=xxx.xxx.xx.x/24
masquerade'
```

Logging

rule ... <log> prefix=”ssh" level="<notice|emergency|alert|crit|error|warning|info|debug>" <audit> limit value="rate/duration"

Port Forwarding (Rich rule & Normal Rule)

```
firewall-cmd --permanent --add-rich-rule='rule
family=ipv4 source address=xxx.xxx.xx.x/24 forward-
port port=xx protocol=tcp to-port=xx to-
addr=xxx.xx.xx.x'
firewall-cmd --permanent --zone=<name> --add-forward-
port=port=<xxxx>:proto=<tcp>[:toport=<xxxx>:toaddr=
<xxx.xxx.xx.x>]
firewall-cmd --<remove-rich-rule=RULE|query-rich-
rule=RULE|list-rich-rules>
```

SELinux

man 8 semanage-fcontext

Install setools-console and list context

```
yum -y install setools-console
seinfo -t | grep <string>
```

SELinux Policy Management port mapping tool

```
semanage port -l
```

```
semanage port -<a|d|m> -t http_port_t -p tcp <88>
```

m=same as removing & adding

```
yum -y install selinux-policy-devel
```

Create or update the manual page index caches

```
mandb
```

Same as apropos, search the manual page names and descriptions:

```
man -k _selinux
```

Generate SELinux man pages sepolicy-manpage

```
sepolicy manpage -a
```

DNS

man unbound.conf

This is the old way of doing things, now handled by nmcli

```
vim /etc/resolv.conf
```

```
host -v -t A example.com
host -v -t AAAA a.root-servers.net
host -v -t A ipa-ca-server0.example.com
host -v -t PTR 172.25.0.10
host -v -t PTR 2001:503:ba3e::2:30
host -v -t <NS|SOA|MX|TXT> example.com
host -v -t SRV _ldap._tcp.server0.example.com
```

Installation

```
yum -y install unbound
systemctl start unbound
systemctl enable unbound
```

Configuration

```
vim /etc/unbound.conf
```

Default is only localhost

```
        interface: 0.0.0.0
```

Default does not accept any connections

```
        access-control: 172.25.0.0/24 allow
```

dot stands for the root domain

```
        forward-zone:
                name: "."
```

Forward query to what DNS

```
                forward-addr: 172.25.254.254
```

Domains not configured with DNSSEC

```
        domain-insecure: example.com
```

```
unbound-checkconf
systemctl restart unbound
firewall-cmd --permanent --add-service=dns
firewall-cmd --reload
unbound-control dump_cache > dump.out
unbound-control load_cache < dump.out
unbound-control flush_zone <example.com>
unbound-control flush <www.example.com>
getent hosts <example.com>
gethostip <example.com>
dig A <example.com>
dig @<dns.example.com> A <www.example.com>
dig +tcp A <example.com>
dig +dnssec DNSKEY <example.com>
```

POSTFIX AS NULL CLIENT

man 5 postconf

/usr/share/doc/postfix-2.10.1/README_FILES/STANDARD_CONFIGURATION_README

```
cp /etc/postfix/main.cf ~/main.cf.orig
```

Needs a change of 6 variables

```
vim /etc/postfix/main.cf
```

Which NIC Postfix listens on for incoming/outgoing messages, can be “all”

```
        inet_interfaces = loopback-only
```

```
        inet_interfaces = all
```

e-mails will appear to come from this domain

```
        myorigin = clientX.example.com
```

Forward all messages to this email server

```
        relayhost = [server.example.com]
```

Which domains the mail server is an end point for, email address to a domain listed here is rejected

```
        mydestination =
```

```
        local_transport = error: local delivery
  disabled
```

Allo relay from these networks

```
        mynetworks = 127.0.0.0/8, [::1]/128
```

```
postfix check
systemctl restart postfix
postconf <-e> 'VAR = VAL'
```

Show only configuration parameters that have explicit name=value settings in main.cf

```
postconf -n
```

```
firewall-cmd --permanent --add-service=smtp
postqueue -<p|f>
mail -s "serverX null client"
student@desktopX.example.com null client test
[ENTER].[ENTER]
```

Postconf Configuration

```
postconf -e "relayhost=[smtp1.example.com]"
postconf -e "inet_interfaces=loopback-only"
postconf -e "mynetworks=127.0.0.0/8 [::1]/128"
postconf -e "myorigin=desktop1.example.com"
postconf -e "mydestination="
postconf -e "local_transport=error: local delivery
disabled"
```

iSCSI

Targets - server creating

man 8 targetcli

```
yum -y install targetcli
```

LVM:

```
fdisk <device> => type 8e
pvcreate <partition>
vgcreate <vgname> <partition>
lvcreate -n <lvname> -L <size> <vgname>
```

Example: lvcreate (-l 100%FREE)

```
fdisk /dev/vdb => type 8e
pvcreate /dev/vdb1
vgcreate iSCSI_vg /dev/vdb1
lvcreate -n disk1_lv -L 100m iSCSI_vg
```

```
targetcli
systemctl start|enable target
cd /backstores
block/ create <block1> /dev/iSCSI_vg/disk1_lv
block/ create <block2> /dev/vdb2
block/ create <file1> /root/disk1_file 100M
cd /iscsi
create iqn.2017-07.com.example:server
cd iqn.2017-07.com.example:server/tpg1
acls/ create iqn.2017-07.com.example:
<client.example.com>
luns/ create /backstores/block/block1
luns/ create /backstores/block/block2
```

```
luns/ create /backstores/fileio/file1
portals/ create 172.25.0.11
```

Or simply portals/ create without IP address

```
exit
firewall-cmd --permanent --add-port=3260/tcp
firewall-cmd --reload
```

Targets – client accessing

/usr/share/doc/iscsi-initiator-utils-6.2.0.873 – Section 7.3 – node.startup

man 8 iscsiadm

```
yum -y install iscsi-initiator-utils
vim /etc/iscsi/initiatorname.iscsi
(InitiatorName=client.example.com)
systemctl restart iscsi
systemctl enable iscsi
iscsiadm -m discovery -t sendtargets -p
172.25.0.11:3260
```

Don’t need port if it’s default

```
iscsiadm -m node -T iqn.2017-07.com.example:server -p
172.25.0.11 -l

iscsiadm -m node -T iqn.2017-05.com.example:server1 -
p 127.25.1.11:3260 -o update -n node.startup -v
automatic
```

```
lsblk --scsi
fdisk /dev/sda
mkfs.xfs/ext4
blkid /dev/sda1 >> /etc/fstab
vim /etc/fstab
UUID=xxxxx-xxxxx-xxxxx /mnt/iscsi xfs _netdev 0 2
```

_netdev is very important and it means mount after networking initialized

```
mount -av
cd /var/lib/iscsi/nodes; ls -lR
iscsiadm -m session -P 3
```

Targets - client disconnecting

```
rm /var/lib/iscsi/nodes/*iqn*
iscsiadm -m node -T iqn.2017-07.com.example:server -p
172.25.0.11 -u
iscsiadm -m node -T iqn.2015-10.com.example:server -p
172.25.0.11 -o delete
systemctl restart iscsi
lsblk
```

NFS

man exports

Server - Insecure

```
yum -y install nfs-utils
systemctl start nfs-server
systemctl enable nfs-server
mkdir /myshare
chown nfsnobody /myshare
vim /etc/exports
        /myshare client.example.com(rw)
        /myshare *.example.com
        /myshare server[0-20].example.com
        /myshare 172.25.0.0/16
        /myshare 172.25.11.10(rw,no_root_squash)
*.example.com(ro)
```

no_root_squash

By default, root on a NFS client is treated as user nfsnobody by the NFS server. That is, if root attempts to access a file on a mounted export, the server will treat it as an access by user nfsnobody instead. This is a security measure that can be problematic in scenarios where the NFS export is used as "/" by diskless clients and root needs to be treated as root.

```
exportfs -r<v>
firewall-cmd --permanent --add-services=nfs
firewall-cmd --reload
showmount -e <server>
```

Client - Insecure

```
yum -y install nfs-utils
systemctl enable nfs
mount server.example.com:/myshare /mnt/nfs
```

```
vim /etc/fstab
        nfserver:/sharename /mountpoint nfs defaults
0 0
```

Server – Secure

```
wget -O /etc/krb5.keytab
http://server.example.com/server.keytab
klist -k; kinit <user>
vim /etc/sysconfig/nfs
        (RPCNFSDARGS="-V 4.2")
systemctl restart nfs-server
systemctl restart nfs-secure-server
systemctl enable nfs-secure-server
vim /etc/exports
        /mysecureshare
client.example.com(sec=krb5p,rw)
```

Uses nfsnobody, needs boolean nfsd_anon_write sec=none Using UID/GUIS linux file permissions [default] sec=sys Kerberos and then Linux file permissions apply sec=krb5 Adds checksums to the data transfers sec=krb5i ADd encryption sec=krb5p

```
exportfs -r<v>
firewall-cmd --permanent --add-services=nfs
firewall-cmd --reload
```

Client – Secure

```
yum -y install nfs-utils
```

Important

```
systemctl start nfs-secure
systemctl enable nfs-secure
```

```
wget -O /etc/krb5.keytab
http://server.example.com/client.keytab
mount -o sec=krb5p,v4.2
server.example.com:/mysecureshare /mnt/nfs
vim /etc/fstab
        serverx:/securenfs /mnt/secureshare nfs
defaults,v4.2,sec=krb5p 0 0
mount -av
```

SELinux

man 8 nfsd_selinux

Context Default:

- ***nfs_t*** – NFS server to access share, both readable and writable
- ***public_content_t*** – NFS and other services to read contents of the share

For writable, change context: *public_content_rw_t*

Doesn't survive FS relabel: chcon -t public_content_t /securenfs/testfile.txt

Booleans

- ***nfs_export_all_ro*** [**default**=on],
- ***nfs_export_all_rw*** [**default**=on],

- ***nfsd_anon_write*** [**default**=off]. It must be enabled for public_content_rw_t e.g.:

setsebool -P nfsd_anon_write=on

SMB

man 5 smb.conf#

Server

```
yum -y install samba samba-client
cp /etc/samba/smb.conf ~/smb.conf.orig
vim /etc/samba/smb.conf
```

Defaults that do not specifically define certain items

```
        [global]
                workgroup=WORKGROUP
```

User-level security where user must be logged in, requires samba password

```
                security=user
                hosts allow=172.25. .example.com
```

e.g. xxx.xx.x.x EXCEPT xxx.xx.x.x, e.g. xxx.xx.x.x/255.0.0.0; can be also **hosts deny=xxx.xx.x.x**

Name of the Share

```
        [myshare]
                path=/sharedpath
                writable=<yes|no>
                write list=<user>
```

Even if writable is no

```
                valid users=<blank>|
<user>|@management|+users
```

By default empty, all users have access to the share. Specifies who can log in to the share.

```
        [homes]
                read only=no
        [printers]
```

```
testparm
groupadd <group>
useradd -s /sbin/nologin -G <group> <user>
```

Change a user's SMB password

```
smbpasswd -<a|x> <user>
```

List all samba accounts configured on the server

```
pdbedit -L
```

```
systemctl reload smb nmb
systemctl enable smb nmb
firewall-cmd --permanent --add-services=samba
firewall-cmd --reload
```

Same as chmod u+rw,g+rws,o+rx /sharedpath `chmod 2775 /sharedpath`

Client – Single User

```
yum -y install cifs-utils
vim /root/credentials.txt
        username=<user>
        password=<password>
```

Same as chmod u+r credentials.txt `chmod 0400 /root/credentials.txt`

By default it uses “sec=ntlmssp `mount -o <username=<user> | credentials=credentials.txt> //server.example.com/<sharename> /mnt/smb`

```
smbclient -L server.example.com
```

Client – Multiuser

```
yum -y install cifs-utils
```

```
useradd <user>
su - <user>
```

Manage NTLM credentials in the keyring) cifscreds <add|update|clear|clearall> -u <user> <server.example.com>

User must exist on the client and have corresponding SMB account on the server

```
mount -o multiuser,sec=ntlmssp,username=
<user>,credentials=<multiuser_file.txt>
//server.example.com/<sharename> /mnt/multiuser
        vim /root/multiuser_file.txt
                username=
<user_with_minimal_permissions_on_the_share>
                password=<password>
        vim /etc/fstab
                //serverX/sambashare /mnt/multiuser
cifs

credentials=/root/multiuser.txt,multiuser,sec=ntlmssp
0 0
mount -av
smbclient -L server.example.com -U <user>
```

SELinux

man 8 samba_selinux

Context:

- ***samba_share_t*** – SMB to access the share
- ***public_content_t & public_content_rw_t*** – accessible by other services as well

Boolean:

- ***smbd_anon_write*** [**default**=off] must be enabled if public_content_rw_t is applied.
- ***boolean for home dirs:***
  - samba_enable_home_dirs [**default**=off] on the server
  - use_samba_home_dirs [**default**=off] on the client

Example: getsebool -a | grep -i <boolean_name>

Permanent change to SE policy file on disk setsebool -P samba_enable_home_dirs=on

Special Permission	**Effect on files**	**Effect on directories**
u+s (suid) **4**xxx	Executes as user who owns, not who runs	---
g+s (sgid) **2**xxx	Executes as group that owns, not who runs	New files have group owner match group owner of the dir
o+t (sticky) **1**xxx	---	Users who can write to the dir can only remove their own files

MARIADB

MariaDB [(none)]> help

```
yum -y groupinstall mariadb mariadb-client
systemctl start mariadb
systemctl enable mariadb
```

Set root passwd,remove anonym,disallow root login,remove testdb
mysql_secure_installation

```
vim /etc/my.cnf
        [mysqld]
```

If blank, only ipv4 is allowed

```
                bind-address <::|0.0.0.0|blank>
```

1=not even localhost can connect,only socket

```
                skip-networking <1|0>
```

Port number 3306 by default

```
                port
```

```
firewall-cmd --permanent --add-rule=mysql
firewall-cmd --reload
mysql -u <root> -h <hostname> -p
create|show|drop database <name>;
use <name>;
```

Managing Users and Access Rights

MariaDB [(none)]> help grant

```
        create user
<user>@'<%|192.168.1.%|localhost>' identified by
'<password>';
mysql -u <user> -h <hostname> -p
        grant select on <database.table> to
<user>@<hostname>;
        grant select on <database.*> to
<user>@<hostname>;
        grant select on <*.*> to <user>@<hostname>;
        grant <create,alter,drop> on <database.*> to
<user>@<hostname>;
        grant all privileges on <*.*> to
<user>@<hostname>;
        revoke <select,update,delete,insert> on
<database.table> from <user>@<hostname>;
        flush privileges;
        show grants for <user>@<hostname>;
        drop user <user>@<hostname>;
```

Backup - Logical

```
mysqldump -u root -p <dbname> > /tmp/dbname.dump
mysqldump -u root -p --<all-databases|add-drop-
tables|no-data|lock-all-tables|add-drop-databases> >
/tmp/all.dump
```

--all-databases will include all user information

Backup - Physical

```
mysqladmin variables | grep datadir
cat /etc/my.cnf | grep -i datadir
df /var/lib/mysql
```

/dev/mapper/vg0-mariadb shows 'vg0' is volume group and 'mariadb' is logical volume name

```
vgdisplay vg0 | grep free
tty0: mysql -u root -p
        tty0: flush tables with read lock;
tty1: lvcreate -L20G -s -n mariadb-backup
/dev/vg0/mariadb
```

-s=snapshot, must be large enough to hold the backup

```
tty0: unlock tables;
mkdir /mnt_snapshot
mount /dev/vg0/mariadb-backup /mnt_snapshot
tar cvzf mariadb_backup.tar.gz
/mnt_snapshot/var/lib/mysql
umount /mnt_snapshot
lvremove /dev/vg0/mariadb-backup
```

Restore - Logical

```
mysql -u root -p <dbname> < /backup/dbname.dump
```

Restore - Physical

```
systemctl stop mariadb
mysqladmin variables | grep datadir
rm -rf /var/lib/mysql/*
tar xvzf mariadb_backup.tar.gz /var/lib/mysql
```

Queries

```
        show databases;
        create table <scientists> (Number int,FirstN
varchar(20),LastN varchar(20));
        select * from product;
        select * from <table1>, <table2> where
‘value1=1’ and ‘value2=2’;
        show tables;
        describe|delete|insert|rename|select|update
<table>;
        insert into <product> (name,price) values
('oracle',1000);
```

Do not insert values into "Auto Increment" fields

```
        delete from <product> where <id=1>;
        delete from <category> where name like
'Memory';
        update <product> set <price=999> where
<id=1>;
        select name,price,stock from product;
        select * from product where price > 90;
        select <field> from <table> where
<field>="x";
        exit;
```

APACHE

http://localhost/manual

yum -y install httpd httpd-manual

```
grep -v '^#' /etc/httpd/conf.d/httpd.conf >
/etc/httpd/conf.d/httpd_without_comments.conf
cp /etc/httpd/conf/httpd.conf ~/httpd.conf.orig
```

Global server configuration

vim /etc/httpd/conf/httpd.conf Where are the config files

```
        ServerRoot "/etc/httpd"
```

Can be 1.2.3.4:80, multiple ports must be specified on separate lines

```
        Listen 80
```

If multiple are present, they will be alphabetically included

```
        Include conf.modules.d/*.conf
```

```
        User apache
        Group apache
        ServerAdmin root@localhost
```

Directives specific to the dir and all descendent dirs

```
        <Directory />
```

.htaccess will not be used

```
                AllowOverride none
```

Refuse to serve content from dir

```
                Require all denied
        </Directory>
```

Where apache looks for files

```
                DocumentRoot "/var/www/html"
```

```
        <Directory  "/var/www/">
                AllowOverride none
                Require all granted
        </Directory>
        <Directory "/var/www/html">
                Options Indexes FollowSymLinks
                AllowOverride none
                Require all granted
        </Directory>
```

If this module is loaded, what happens

```
        <IfModule dir_module>
```

This file will be used when the directory is requested

```
                DirectoryIndex index.html
        </IfModule>
```

Same as directory but for file wildcards

```
<Files ".ht*">
                Require all denied
        </Files>
```

IT will go for /etc/httpd/logs/error_log, which is symlink to /var/log/httpd/error_log

```
        ErrorLog "logs/error_log"
        LogLevel warn
        CustomLog "logs/access_log" combined
```

Can be disabled by AddDefaultCharset Off

```
        AddDefaultCharset UTF-8
```

Same as Regular include

```
        IncludeOptional conf.d/*.conf (same as
regular include)
```

Validate the config files `httpd -t`

```
systemctl enable httpd
systemctl start httpd
firewall-cmd --permanent --add-service=http --add-
service=https
firewall-cmd --reload
semanage port -l | grep '^http_'
```

New DocumentRoot for group 'webmasters'

Same as chmod u+rw, g+rws, o+rx /new/web

```
mkdir -p -m 2775 /new/web
```

```
groupadd webmasters
chgrp webmasters /new/web
chmod 2775 /new/web
```

X=Keeps executable settings,directories allow directory search,**x**=executable

```
setfacl -R -m g:webmasters:rwX /new/web
setfacl -R -m d:g:webmasters:rwX /new/web
```

Rules are already in place to relabel /srv/*/www

```
semanage fcontext -a -t httpd_sys_content_t
"/new/web(/.*)?"
```

Resets the context on the files AFTER you create them

```
restorecon -Rv /new/web
```

```
systemctl reload httpd
```

Private directory protected by password

```
<Directory /var/www/private>
```

Set basic authentication

```
AuthType basic
AuthName "This site is protected. Enter password:"
```

Specifies the file with user/passwd

```
AuthUserFile /etc/httpd/conf/userpasswords
Require user user1
```

Or simply valid-user for anyone in the userpasswords file

```
</Directory>
htpasswd -bc /etc/httpd/conf/userpasswords user1
p4ssw0rd
chmod 0640 /etc/httpd/conf/userpasswords
```

```
chgrp apache /etc/httpd/conf/userpasswords
```

Together with AuthUserFile, you can use AuthGroupFile and Require group. Content of the group file is: `cat /etc/httpd/conf/grouppasswords: groupname: user1 user2 user3`. These users must be in userpasswords file

Virtual Hosts

```
vim /etc/httpd/conf.d/00-site1.conf
```

This block provides access to Document Root further down

```
        <Directory /srv/site1/www>
                Require all granted
                AllowOverride none
        </Directory>
```

This block must be considered for all connections on 192.168.0.1:80, can be *default*:80 or *:80 which will ALWAYS match for regular http traffic, effectively disabling the main server config from ever being used on port 80.

```
        <VirtualHost 192.168.0.1:80>
```

Only applies for within this Virtual Host

```
                DocumentRoot /srv/site1/www
```

Name-based virtual hosting, if multiple virtual hosts are defined, the one where hostname matches this will be used, it is best to always explicitly use this. It doesn't need to exist, if you need "match anything" – e.g. all other domains types of VirtualHosts

```
                   ServerName site1.example.com[:80]
```

If the virtual host needs to be used for more than one domain name, wildcards can be used e.g. *.example.com

```
                   ServerAlias site1
                   ServerAdmin root@site1.example.com
                   ErrorLog "logs/site1_error_log"
                   CustomLog "logs/site1_access_log"
combined
        </VirtualHost>
```

```
httpd —D DUMP_VHOSTS
semanage fcontext -a -t httpd_sys_content_t
"/srv/site1/www(/.*)?"
restorecon -Rv /srv/site1/www
```

If there are multiple catch-all VirtualHosts, they will be executed alphabetically (e.g. 00-default.conf,default.conf,vhost.conf).

How the server selects the proper name-based virtual host? When a request arrives, the server will find the most specific virtual host argument based on IP/port used by the request. If there is more than one containing the best-match, Apache will further compare the ServerName and ServerAlias directives to the server name present in the request. If no matching ServerName/ServerAlias is found in the set of virtual hosts, then the first listed virtual host that matches will be used.

Any request that does not match existing virtual host is handled by the global server configuration /etc/httpd/conf/httpd.conf, regardless of hostname/ServerName. When you add virtual host to an existing server and the virtual host match preexisting IP/port, request will now be handled virtual host. In this case, it is wise to create default virtual host with ServerName matching the base server.

Access Control Directives

<RequireAll></RequireAll> – none must fail and at least one must succeed ***<RequireAny></RequireAny>*** – one or more must succeed ***<RequireNone></RequireNone>*** – none must succeed

If it is not enclosed in directives, it is automatically <RequireAny>

Examples

1. Address is an IP, partial IP, network/mask, network/CIDR, ipv4/ipv6

```
<RequireAll>
      Require all granted
Require not ip 10.252.46.125
</RequireAll>
```

2. Address is FQDN or part of it, multiple may be provided

```
<RequireAll>
Require all granted
Require not ip 192.168.2.1
Require not host phishers.example.com
moreidiots.example
```

```
Require not host gov
</RequireAll>
```

3.

```
Require all denied
Require local
```

4. Only allows specific hostname

```
Require host test.example.com
```

5. Can be username / UID

```
Require User John
```

6. Can be groupname /GID

```
Require not user badjohn
```

7.

```
Require ip 192.168.0 15.2
```

SSL/TLS

```
yum -y install crypto-utils mod_ssl
genkey <www.example.com>
cp /etc/httpd/conf.d/ssl.conf ~/ssl.conf.orig
grep -v '^#' /etc/httpd/conf.d/ssl.conf >
/etc/httpd/conf.d/ssl_without_comments.conf
vim /etc/httpd/conf.d/ssl.conf
        Listen 443 https
```

If the private key uses passphrase

```
        SSLPassPhraseDialog exec:/usr/libexec/httpd-
ssl-pass-dialog
        <VirtualHost _default_:443>
                SSLEngine on
```

ServerName www.example.com[:443] Public Key

```
                SSLCertificateFile
/etc/pki/tls/certs/www.example.com.crt
```

Private Key

```
                SSLCertificateKeyFile
/etc/pki/tls/certs/www.example.com.key
```

Copy of all CA Certificates

```
                SSLCertificateChainFile
/etc/pki/tls/certs/example-ca.crt
```

```
        DocumentRoot /var/www/html
        </VirtualHost>
```

This is the Default

```
ls -Zd /etc/pki/tls/
semanage fcontext -a -t cert_t "/etc/pki/tls(/.*)?"
restorecon -Rv /etc/pki/tls/
```

Same as chmod u+rw *.key

```
chmod 0600 /etc/pki/tls/private/*.key
```

same as chmod u+rw,g+r,o+r *.crt

```
chmod 0644 /etc/pki/tls/certs/*.crt
```

HSTS - strict transport security

```
        <VirtualHost *:80>
        ServerName...;ServerAlias...;DocumentRoot...
        Header always set Strict-Transport-Security
"max_age=15768000"
        RewriteEngine on
        RewriteRule ^(/.*)$ https://%{HTTP_POST}$1
[redirect=301]
        <VirtualHost>
```

Dynamic content

1. **CGI**

```
        vim /etc/httpd/conf/httpd.conf
```

First parameter is part of the URL, second is the location of the script.

```
                ScriptAlias /cgi-bin/ "/var/www/cgi-
bin/"
```

```
<Directory /var/www/html>
        Options none
        Require all granted
</Directory>
```

SELinux fcontext: httpd_sys_script_exec_t, httpd_enable_cgi

2. **PHP**

```
        yum -y install mod_php php php-mysql
                <FilesMatch \.php$>
                        SetHandler application/x-
httpd-php
                </FilesMatch>
                DirectoryIndex index.php
```

3. **Python**

```
        yum -y install mod_wsgi
        vim /etc/httpd/conf/httpd.conf
```

A request for www.example.com/myapp will cause the server to run the WSGI application defined in /srv/my.py

```
        WSGIScriptAlias /myapp "/srv/my.py"
```

SELinux fcontext: httpd_sys_content_t

SELinux

man 8 httpd_selinux

```
semanage port -l | grep '^http_'
```

Non-Standard HTTP Ports

```
semanage port -a -t http_port_t -p tcp 88
```

```
semanage fcontext -a -t httpd_sys_content_t
"/srv/site1/www(/.*)?"
```

Not before files are present

```
restorecon -Rv /srv/site1/www
```

Context:

httpd_sys_content_t – Dirs where Apache is allowed to access
httpd_sys_content_rw_t – Dirs where Apache is allowed to read/write
httpd_sys_script_exec_t – dirs that contain executable scripts **cert_t** – Dirs where Apache is allowed to read SSL certificates

Booleans:

httpd_unified *[default=off]* – Simplified/unified policy when turned on

httpd_enable_cgi *[default=on]* – Allowed to run scripts

httpd_tty_comm *[default=off]* – Apache is allowed to access TTY, switch on when using private key with passkey

httpd_can_network_connect_db *[default=off]* – If the database is on remote host

httpd_can_network_connect *[default=off]* – If the known port number is used for db connection

httpd_anon_write *[off]*, **httpd_sys_script_anon_write** *[off]* – If directory that is using public_content_rw_t is being used by Apache

SHELL ENVIRONMENT

Global

```
        /etc/profile
        /etc/profile.d/*.sh
        /etc/bashrc
```

User

```
          ~/.bash_profile, .bash_login, .profile
          ~/.bashrc
```

1. **Profiles** are for setting and exporting of environment variables, as well as running commands that should only be run upon login. Usually, profiles are only executed in a login shell, whereas RCs are executed every time a shell is created, login or non-login
2. RCs are for running commands, setting aliases, defining functions and other settings that cannot be exported to sub-shells.

Supplied MYVAR are marked for automatic export to the environment of subsequently executed commands.

```
export MYVAR
alias
unalias
function () {...}
set
unset
```

Bash

```
chmod +x script.sh
```

```
$VARIABLENAME vs. ${VARIABLENAME}
      $FIRST_$LAST    = $FIRST_ + $LAST
      ${FIRST}_$LAST = $FIRST +_ + $LAST
`CMD` == $(CMD)
$[<ARITHEMTIC EXPRESSION>]
FOR <VARIABLE> in <LIST>; do
      <COMMAND>
...
      <COMMAND> referencing <VARIABLE>
DONE
```

Example:

```
cat file
        peter
        john
vim script.sh

#!/bin/bash
file=$(cat $1)
for i in $file; do
        echo $i
done
```

Troubleshooting:

```
bash -x <SCRIPT> or 'set -x' ... 'set +x'
bash -v <SCRIPT> or 'set -v' ... 'set +v'
```

- *$0* = script name itself
- *$1* = first argument of the script
- *$*, $@* = all arguments
- *$#* = number of arguments
- *$?* = exit status/code (exit 0 -> exit 255)

Comparison:

["$A" -eq "$B"]; ... $?

- *'eq' or '='* = equal
- *'ne' or '!='* = not equal
- *'gt'* = greater than
- *'ge'* = greater/equal than
- *'lt'* = less than
- *'le'* = less/equal than
- *'z'* = string is null
- *'n'* = string is not null
- *'b'* = file exists & block special
- *'c'* = file exists & character special
- *'d'* = is directory
- *'e'* = exists
- *'f'* = is regular file
- *'L'* = is symbolic link
- *'r'* = read permission granted
- *'s'* = non-zero size
- *'w'* = write permission granted
- *'x'* = execute permission granted
- *'ef'* = same device & inode
- *'nt'* = newer modification date
- *'ot'* = older modification date
- *&&* = AND
- *//* = OR

```
if <CONDITION>; then
      <CMD>
elif <STATEMENT>
else <STATEMENT>
fi
```

```
case <VALUE> in
      <PATTERN1>) <STATEMENT>;;
      <PATTERN2>) <STATEMENT>;;
      <PATTERN3>) <STATEMENT>;;
      <*>) ;;
esac
```

Exercises

dbbackup

```
vim dbbackup
chmod +x dbbbackup
```

```
#!/bin/bash
#RHCE page 341, guided exercise

#Variables
DBUSER=root
FMTOPTIONS='--skip-column-names -E'
COMMAND='SHOW DATABASES'
BACKUPDIR=/dbbackup

#Backup non-system databases
for DBNAME in $(mysql $FMOPTIONS -u $DBUSER -e
"$COMMAND" | grep -v ^* | grep -v information_schema
| grep -v performance_schema); do
        echo "Backing up \"$DBNAME\""
        mysqldump -u $DBUSER $DBNAME >
$BACKUPDIR/$DBNAME.dump
done

#Add up size of all database dumps
for DBDUMP in $BACKUPDIR/*; do
```

```
        SIZE=$(stat --printf "%s\n" $DBDUMP)
        TOTAL=$[ $TOTAL + $SIZE]
done

#Report name, size, and percentage of total for each
database dump
echo
for DBDUMP in $BACKUPDIR/*; do
        SIZE=$(stat --print "%s\n" $DBDUMP)
        echo "$DBDUMP,$SIZE,$[ 100 * $SIZE / $TOTAL
]%"
done
```

mkaccounts.orig

```
vim mkaccounts.orig
chmod +x mkaccounts.orig
```

```
#!/bin/bash
#RHCE page 347, lab exercise

#Variables
NEWUSERSFILE=/tmp/support/newusers

#Loop
for ENTRY in $(cat $NEWUSERSFILE); do
        #Extract first, last and tier fields
        FIRSTNAME=$(echo $ENTRY | cut -d: -f1)
        LASTNAME=$(echo $ENTRY | cut -d: -f2)
        TIER=$(echo $ENTRY | cut -d: -f4)
        #Make account name
        FIRSTINITIAL=$(echo $FIRSTNAME | cut -c 1 |
tr 'A-Z' 'a-z')
        LOWERLASTNAME=$(echo $LASTNAME | tr 'A-Z' 'a-
z')
        ACCTNAME=$$FIRSTINITIAL$LOWERLASTNAME
        #Create account
        useradd $ACCTNAME -c "$FIRSTNAME $LASTNAME"
done
TOTAL=$(cat $NEWUSERSFILE | wc -l)
TIER1COUNT=$(grep -c :1$ $NEWUSERSFILE)
TIER2COUNT=$(grep -c :2$ $NEWUSERSFILE)
TIER3COUNT=$(grep -c :3$ $NEWUSERSFILE)
TIER1PCT=$[ $TIER1COUNT * 100 / $TOTAL ]
TIER2PCT=$[ $TIER2COUNT * 100 / $TOTAL ]
TIER3PCT=$[ $TIER3COUNT * 100 / $TOTAL ]

#Print the report
echo "\"Tier 1\",\"$TIER1COUNT\",\"$TIER1PCT%\""
echo "\"Tier 2\",\"$TIER2COUNT\",\"$TIER2PCT%\""
echo "\"Tier 3\",\"$TIER3COUNT\",\"$TIER3PCT%\""
```

mkvhost

```
vim mkvhost
chmod +x mkvhost
```

```
#!/bin/bash
#RHCE page 363, guided exercise

#Variables
VHOSTNAME=$1
TIER=$2
HTTPDCONF=/etc/httpd/conf/httpd.conf
VHOSTCONFDIR=/etc/httpd/conf.vhost.d
DEFHOSTCONFFILE=$VHOSTCONFDIR/00-default-vhost.conf
VHOSTCONFFILE=$VHOSTCONFDIR/$VHOSTNAME.conf
WWWROOT=/srv
```

```
DEFVHOSTDOCROOT=$WWWROOT/default/www
VHOSTDOCROOT=$WWWROOT/$VHOSTNAME/www

#Check arguments
if [ "$VHOSTNAME" = '' ] || [ "$TIER" = '' ]; then
        echo "Usage: $0 VHOSTNAME TIER"
        exit 1
else

#Set support email address
   case $TIER in
        1)VHOSTADMIN='basic_support@example.com'
          ;;
        2)VHOSTADMIN='business_support@example.com'
          ;;
        3)VHOSTADMIN='enterprise_support@example.com'
          ;;
        *)echo "Invalid tier specified."
          exit 1
          ;;
   esac
fi

#Create conf directory one time if non-existent
if [ ! -d $VHOSTCONFDIR ]; then
        mkdir $VHOSTCONFDIR
        if [ $? -ne 0 ]; then
                echo "ERROR: Failed creating
$VHOSTCONFDIR."
                exit 1
        fi
fi

#Add include one time if missing
grep -q '^IncludeOptional conf\.vhosts\.d/\*\.conf$'
$HTTPDCONF
if [ $? -ne 0 ]; then
        #Backup before modifying
        cp -a $HTTPDCONF $HTTPDCONF.orig
        echo "IncludeOptional conf.vhosts.d/*.conf"
>> $HTTPDCONF
        if [ $? -ne 0 ]; then
                echo "ERROR: Failed adding include
directive."
                exit 1
        fi
fi

#Check for default virtual host
if [ ! -f $DEFVHOSTCONFFILE ]; then
        cat <<DEFCONFEOF > $DEFVHOSTCONFFILE
<VirtualHost _default_:80>
        DocumentRoot $DEFVHOSTDOCROOT
        CustomLog "logs/default-vhost.log" combined
</VirtualHost>
<Directory $DEFVHOSTDOCROOT>
        Require all granted
</Directory>
DEFCONFEOF
fi

if [ ! -d $DEFVHOSTDOCROOT ]; then
        mkdir -p $DEFVHOSTDOCROOT
        restorecon -Rv /srv/
fi

#Check for virtual host conflict
if [ -f $VHOSTCONFFILE ]; then
        echo "ERROR: $VHOSTCONFFILE already exists."
        exit 1
elif [ -d $VHOSTDOCROOT ]; then
        echo "ERROR: $VHOSTDOCROOT already exists."
        exit 1
else
        cat <<CONFEOF > $VHOSTCONFFILE
<Directory $VHOSTDOCROOT>
        Require all granted
        AllowOverride None
```

```
</Directory>
<VirtualHost *:80>
        DocumentRoot $VHOSTDOCROOT
        ServerName $VHOSTNAME
        ServerAdmin $VHOSTADMIN
        ErrorLog "logs/${VHOSTNAME}_error_log"
        CustomLog "logs/${VHOSTNAME}_access_log"
common
</VirtualHost>
CONFEOF
        mkdir -p $VHOSTDOCROOT
        restorecon -Rv $WWWROOT
fi

#Check config and reload
apachectl configtest &> /dev/null
if [ $? -eq 0 ]; then
        systemctl reload httpd &> /dev/null
else
        echo "ERROR: Config error."
        exit 1
fi
```

mkaccounts

```
vi mkaccounts
chmod +x mkaccounts
```

```
#!/bin/bash
#RHCE page 370, lab exercise

#Variables
OPTION=$1
```

```
NEWUSERSFILE=/tmp/support/newusers

case $OPTION in
        '')
            ;;
        -v) VERBOSE=y
            ;;
        -h) echo "Usage: $0 [-h|-v]"
            echo
            exit
            ;;
         *) echo "Usage: $0 [-h|-v]"
            echo
            exit 1
            ;;
esac

#Test for dups and conflicts
ACCTEXIST=''
ACCTEXISTNAME=''
if [ $? -eq 0 ]; then
        ACCTEXIST=y
        ACCTEXISTNAME="$(grep ^$ACCTNAME: /etc/passwd
| cut -f5 -d:)"
fi
if [ "$ACCTEXIST" = 'y' ] && [ "$ACCTEXISTNAME" =
"$FIRSTNAME $LASTNAME" ]; then
        echo "Skipping $ACCTNAME. Duplicate found."
elif ["$ACCTEXIST" = 'y' ]; then
        echo "Skipping $ACCTNAME. Conflict found."
else    useradd $ACCTNAME -c "$FIRSTNAME $LASTNAME"
        if [ "$VERBOSE" = 'y' ]; then
        echo "Added $ACCTNAME."
        fi
fi
#Loop
for ENTRY in $(cat $NEWUSERSFILE); do
        #Extract first, last and tier fields
        FIRSTNAME=$(echo $ENTRY | cut -d: -f1)
        LASTNAME=$(echo $ENTRY | cut -d: -f2)
        TIER=$(echo $ENTRY | cut -d: -f4)
        #Make account name
        FIRSTINITIAL=$(echo $FIRSTNAME | cut -c 1 |
tr 'A-Z' 'a-z')
        LOWERLASTNAME=$(echo $LASTNAME | tr 'A-Z' 'a-
z')
        ACCTNAME=$$FIRSTINITIAL$LOWERLASTNAME
        #Create account
        useradd $ACCTNAME -c "$FIRSTNAME $LASTNAME"
done
TOTAL=$(cat $NEWUSERSFILE | wc -l)
TIER1COUNT=$(grep -c :1$ $NEWUSERSFILE)
TIER2COUNT=$(grep -c :2$ $NEWUSERSFILE)
TIER3COUNT=$(grep -c :3$ $NEWUSERSFILE)
TIER1PCT=$[ $TIER1COUNT * 100 / $TOTAL ]
TIER2PCT=$[ $TIER2COUNT * 100 / $TOTAL ]
TIER3PCT=$[ $TIER3COUNT * 100 / $TOTAL ]

#Print the report
echo "\"Tier 1\",\"$TIER1COUNT\",\"$TIER1PCT%\""
echo "\"Tier 2\",\"$TIER2COUNT\",\"$TIER2PCT%\""
echo "\"Tier 3\",\"$TIER3COUNT\",\"$TIER3PCT%\""
```

myusers

```
vi myusers
chmod +x myusers
```

```
#!/bin/bash
#RHCE page 419, comprehensive review lab

if [ $# -eq 0 ]; then
        echo "$(basename $0) userlist"
        echo "$(basename $0) userinfo <USERNAME>"
fi

case $1 in
        userlist) grep -v ':/sbin/nologin$'
/etc/passwd | cut -d: -f1 | sort
                  ;;
        userinfo) if [ "$2" == "" ]; then
                        echo "Please specify a
username"
                        exit 132
                  fi
                  if ! getent passwd $2 &> /dev/null;
then
                        echo "Invalid user"
                        exit
                  fi
                  getent passwd $2 | cut -d: -f7
                  ;;
        *) exit
           ;;
esac
```