

```
I. vim /etc/yum.repos.d/rhce.repo
[RHCE_RHEL7]
name=RHCE_RHEL7
baseurl=http://.../.../...
enabled=1
gpgcheck=0
yum repolist
```

```
II. a/ allow SSH for xyz.com and deny SSH to all the others:
      vim /etc/hosts.allow -> sshd: .xyz.com
      vim /etc/hosts.deny -> sshd: ALL
    b/ allow SSH for only specific IP and block all the others:
      vim /etc/hosts.deny -> sshd: ALL EXCEPT 192.168.0.1
    c/ 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
    d/ 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
    e/ 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
```

```
III. Recover root passwd: reboot;e;linux16...rd.break;ctrl+x;mount -o remount,rw /sysroot;
      chroot /sysroot;passwd root;touch /.autorelabel;exit;exit;
```

```
systemctl --failed --type=service
systemctl show <unit>
systemctl status <-l> <unit>
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>
```

```
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.xx.x/24 gw4
xxx.xxx.xx.x
nmcli con <up|down> <name>
nmcli dev status
nmcli dev dis <eth0>
nmcli con mod <name> +ipv4.dns xxx.xxx.xx.x
      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 (-another utility to investigate sockets)
```

```
nmcli con add con-name <name> type ethernet ifname <eth0> ip6 xxxx:xxxx:xxx: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 (-print network connections, routing tables, interface statistics, masquerade
connections, and multicast memberships)
```

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

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

```
a/ Understand 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
```

b/ Rules

```
firewall-cmd --<get-default-zone|set-default-zone|get-zones|get-services|get-active-
zones|list-all>
firewall-cmd --<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
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 --permanent --zone=<name> --add-masquerade
firewall-cmd --permanent --zone=<name> --add-rich-rule='rule family=ipv4 source
address=xxx.xxx.xx.x/24 masquerade'
```

c/ Port forwarding

```
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>]
```

c/ SELinux #man 8 semanage-fcontext#

```
semanage port -l (-SELinux Policy Management port mapping tool)
semanage port -a|d|m> -t http port_t -p tcp <88>
```

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

```
mandb (-create or update the manual page index caches)
man -k _selinux (-same as apropos, search the manual page names and descriptions)
sepolycy manpage -a (-generate SELinux man pages sepolycy-manpage)
```

#7 DNS #man unbound.conf#

```
vim /etc/resolv.conf (-this is the old way of doing things, now handled by nmcli)
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|2001:503:ba3e::2:30>
host -v -t <NS|SOA|MX|TXT> example.com
host -v -t SRV ldap. tcp.server0.example.com
```

```
yum -y install unbound
```

```
systemctl start unbound
systemctl enable unbound
cp /etc/unbound.conf ~/unbound.conf.orig
vim /etc/unbound.conf
    interface: 0.0.0.0 (-default is only localhost)
    access-control: 172.25.0.0/24 allow (-default does not accept any connections)
    forward-zone:
        name: "." (-dot stands for the root domain)
        forward-addr: 172.25.254.254 (-forward query to what DNS?)
    domain-insecure: example.com (-domains not configured with DNSSEC)
```

```
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>
```

#8 POSTFIX AS NULL CLIENT #man 5 postfixconf#

```
cp /etc/postfix/main.cf ~/main.cf.orig
vim /etc/postfix/main.cf (-needs a change of 6 variables)
    inet_interfaces = loopback-only (-which NIC Postfix listens on for
incoming/outgoing messages)
    myorigin = example.com (-e-mails will appear to come from this domain)
```

```

    relayhost = [server.example.com]    (-forward all messages to this mail server)
    mydestination =                      (-which domains the mail server is an end point
for, mail address to a domain listed here is rejected)
    local_transport = error: local delivery disabled
    mynetworks = 127.0.0.0/8, [::1]/128 (-allow relay from these networks)
postfix check
systemctl restart postfix
postconf -e 'VAR = VAL'
postconf -n (-show only configuration parameters that have explicit name=value settings in
main.cf)
firewall-cmd --permanent --add-service=smtp
postqueue -c
mail -s "serverX null client" student@desktopX.example.com null client test

```

#9

iSCSI

a/ Targets - server creating

```

yum -y install targetcli
LVM: fdisk <device> => type 8e; pvcreate <partition>; vgcreate <vgname> <partition>;
lvcreate -n <lvname> -L <size> <vgname>
    fdisk /dev/vdb => type 8e; pvcreate /dev/vdb1; vgcreate iSCSI_vg /dev/vdb1;
lvcreate -n disk1_lv -L 100m iSCSI_vg
targetcli
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.2015-10.com.example:server
cd iqn.2015-10.com.example:server/tpg1
acls/ create iqn.2015-10.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
systemctl enable target

```

b/ Targets - client accessing

```

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.2015-10.com.example:server -p 172.25.0.11 -l
lsblk --scsi
blkid
fdisk, mkfs.xfs/ext4/etc. ...
vim /etc/fstab
    UUID=xxxxxx-xxxxxx-xxxxxx /mnt/iscsi xfs _netdev 0 2 (-netdev is very important
and it means mount after networking initialized)
mount -av
iscsiadm -m session -P 3
ls -lR /var/lib/iscsi/nodes

```

c/ Targets - client disconnecting

```

iscsiadm -m node -T iqn.2015-10.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
lsblk
systemctl restart iscsi

```

a/ *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)
```

```
exportfs -r<v>
```

```
firewall-cmd --permanent --add-services=nfs
```

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

```
showmount -e <server>
```

b/ *Client - insecure*

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

```
systemctl enable nfs
```

```
mount server.example.com:/myshare /mnt/nfs
```

```
vim /etc/fstab
```

```
    nfsserver:/sharename /mountpoint nfs defaults 0 0
```

c/ *Server - secure*

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

```
wget -O /etc/krb5.keytab http://xxxxxxxxxx
```

```
klist -k; kinit <user>
```

```
vim /etc/sysconfig/nfs (RPCNFSDARGS="-V 4.2")
```

```
systemctl restart nfs-server nfs-secure-server (-important!)
```

```
systemctl enable nfs-server nfs-secure-server (-important!)
```

```
vim /etc/exports
```

```
    /myseureshare client.example.com(sec=krb5p,rw)
```

```
        sec=none      (-uses nfsnobody, needs boolean nfsd_anon_write)
```

```
        sec=sys       (-using UID/GUIS linux file permissions) [default]
```

```
        sec=krb5      (-kerberos and then linux file permissions apply)
```

```
        sec=krb5i     (-adds checksums to the data transfers)
```

```
        sec=krb5p     (-adds encryption)
```

```
exportfs -r<v>
```

```
firewall-cmd --permanent --add-services=nfs
```

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

d/ *Client - secure*

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

```
systemctl start nfs-secure (-important!)
```

```
systemctl enable nfs-secure (-important!)
```

```
wget -O /etc/krb5.keytab http://xxxxxxxxxx
```

```
mount -o sec=krb5p,v4.2 server.example.com:/myseureshare /mnt/nfs
```

```
vim /etc/fstab
```

```
    serverx:/secrenfs /mnt/seureshare nfs defaults,v4.2,sec=krb5p 0 0
```

```
mount -av
```

e/

SELinux #man 8 nfsd_selinux#

```
context:  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 access, change context:
```

```
    public_content_rw_t
```

```
booleans: nfs_export_all_ro [default=on],
```

```
          nfs_export_all_rw [default=on],
```

```
          nfsd_anon_write [default=off] must be enabled for public_content_rw_t
```

```
e.g. setsebool -P nfsd_anon_write=on
```

a/ Server

```
yum -y install samba samba-client
```

```
cp /etc/samba/smb.conf ~/smb.conf.orig
```

```
vim /etc/samba/smb.conf
```

```
[global]                                (-defaults that do not specifically define certain items)
```

```
workgroup=WORKGROUP
```

```
security=user                          (-user-level security where user must be logged in, requires
```

```
samba password)
```

```
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.)
```

```
[myshare]                               (-name of the share)
```

```
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
```

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

```
smbpasswd -<a|x> <user> (-change a user's SMB password)
```

```
systemctl reload smb nmb
```

```
systemctl enable smb nmb
```

```
firewall-cmd --permanent --add-services=samba
```

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

```
chmod 2775 /sharedpath (-same as chmod u+rw,g+rws,o+rx /sharedpath)
```

b/ Client - singleuser

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

```
vim /root/credentials.txt
```

```
username=<user>
```

```
password=<password>
```

```
chmod 0400 /root/credentials.txt (-same as chmod u+r credentials.txt)
```

```
mount -o <username=<user>|credentials=credentials.txt> //server.example.com/<sharename>
```

```
/mnt/smb
```

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

c/ Client - multiuser

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

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

```
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=<password1>
```

```
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>
```

d/

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
```

```
e.g. getsebool -a | grep -i <boolean_name>
```

```
setsebool -P samba_enable_home_dirs=on (-permanent change to SE policy file on
```

```
disk)
```

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

```
#12 MARIADB #MariaDB [(none)]> help#
```

```
yum -y groupinstall mariadb mariadb-client
```

```
systemctl start mariadb
```

```
systemctl enable mariadb
```

```
mysql_secure_installation (-set root passwd,remove anonym,disallow root login,remove testdb)
```

```
vim /etc/my.cnf
```

```
[mysqld]
```

```
bind-address <::|0.0.0.0|blank> (-if blank, only ipv4 is allowed)
```

```
skip-networking <1=not even localhost can connect,only socket|0>
```

```
port (-port number 3306 by default)
```

```
firewall-cmd --permanent --add-rule=mysql
```

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

```
mysql -u <root> -h <hostname> -p
```

```
create|show|drop database <name>;
```

```
use <name>;
```

```
a/ 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>;
```

```
b/ 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
```

```
c/ Backup - physical
```

```
mysqldadmin 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)
```

```
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
```

d/ *Restore - logical*

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

e/ *Restore - physical*

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

f/ *Queries*

```
show databases;
create table <scientists> (Number int,FirstN varchar(20),LastN varchar(20));
select * from product;
show tables;
describe|delete|insert|rename|select|update <table>;
insert into <product> (name,price) values ('oracle',1000);
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>="whatever";
exit;
```

#13 APACHE #http://localhost/manual#

yum -y install httpd httpd-manual elinks

```
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
```

```
vim /etc/httpd/conf/httpd.conf (-global server configuration)
```

```
ServerRoot "/etc/httpd" (-where are the config files)
```

```
Listen 80 (-can be 1.2.3.4:80, multiple ports must be specified on
separate lines)
```

```
Include conf.modules.d/*.conf (-if multiple are present, they will be alphabetically
included)
```

```
User apache
```

```
Group apache
```

```
ServerAdmin root@localhost
```

```
<Directory /> (-directives specific to the dir and all descendent dirs)
```

```
AllowOverride none (-.htaccess will not be used)
```

```
Require all denied (-refuse to serve content from dir)
```

```
</Directory>
```

```
DocumentRoot "/var/www/html" (-where apache looks for HTML files)
```

```
<Directory "/var/www/">
```

```
AllowOverride none
```

```
Require all granted
```

```
</Directory>
```

```
<Directory "/var/www/html">
```

```
Options Indexes FollowSymLinks
```

```
AllowOverride none
```

```
Require all granted
```

```
</Directory>
```

```
<IfModule dir_module>
```

(-if this module is loaded, what happens)

```
DirectoryIndex index.html (-this file will be used when the directory is
requested)
```

```
</IfModule>
```

```
<Files ".ht*">
```

(-same as directory, but for file wildcards)

```
Require all denied
```

```
</Files>
```

```
ErrorLog "logs/error_log"
```

(-it will go to /etc/httpd/logs/error_log, which is
symlink to /var/log/httpd/error_log)


```

LogLevel warn
CustomLog "logs/access_log" combined
AddDefaultCharset UTF-8          (-can be disabled by AddDefaultCharset Off)
IncludeOptional conf.d/*.conf    (-same as regular include)

httpd -t (-this is to validate the config files)
systemctl enable httpd
systemctl start httpd
firewall-cmd --permanent --add-service=http --add-service=https
firewall-cmd --reload

a/ New DocumentRoot for group 'webmasters'
    mkdir -p -m 2775 /new/web (-same as chmod u+rw,g+rws,o+rx /new/web)
    groupadd webmasters
    chgrp webmasters /new/web
    chmod 2775 /new/web
    setfacl -R -m g:webmasters:rwX /new/web (X=retain executable
settings,directories allow directory search,x=executable)
    setfacl -R -m d:g:webmasters:rwX /new/web
    semanage fcontext -a -t httpd_sys_content_t "/new/web(/.*)?"
    restorecon -Rv /new/web
    systemctl reload httpd

b/ Private directory protected by password
<Directory /var/www/private>
    AllowOverride AuthConfig
    AuthType basic (-set basic authentication)
    AuthName "This site is protected. Enter password:"
    AuthUserFile /etc/httpd/conf/userpasswords (specifies the file with user/passwd)
    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)

c/ Virtual hosts
vim /etc/httpd/conf.d/00-sitel.conf
    <Directory /srv/sitel/www>          (-this block provides access to document root
further down)
        Require all granted
        AllowOverride none
    </Directory>
    <VirtualHost 192.168.0.1:80>          (-this block must be considered for all connections
on 192.168.0.1:80, can be _default_:80 or *:80)
        DocumentRoot /srv/sitel/www    (-only applies for within this virtual host)
        ServerName sitel.example.com:80 (-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)
        ServerAlias sitel              (-if the virtual host needs to be used for more
than one domain name, wildcards can be used e.g. *.example.com)
        ServerAdmin root@sitel.example.com
        ErrorLog "logs/sitel_error_log"
        CustomLog "logs/sitel_access_log" combined
    </VirtualHost>

httpd -D DUMP_VHOSTS
(-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.)

c/ 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>

e.g.

I. <RequireAll>

Require all granted

Require not ip 10.252.46.125 (-address is an IP, partial IP, network/mask, network/CIDR, ipv4/ipv6)

</RequireAll>

II. <RequireAll>

Require all granted

Require not ip 192.168.2.1

Require not host phishers.example.com moreidiots.example (-address is FQDN or part of it, multiple may be provided)

Require not host gov

</RequireAll>

III. Require all denied

Require local

IV. Require host test.example.com (-to only allow specific hostname)

V. Require user john (-can be username/UID)

VI. Require not user badjohn (-can be groupname/GID)

VII. Require ip 192.168.0 15.2

d/ 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

SSLPassPhraseDialog exec:/usr/libexec/httpd-ssl-pass-dialog (-if the private key uses passphrase)

<VirtualHost _default_:443>

SSLEngine on

SSLProtocol all -SSLv2

SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5

(ServerName www.example.com:443)

(SSLHonorCipherOrder On)

SSLCertificateFile /etc/pki/tls/certs/www.example.com.crt (-public key)

SSLCertificateKeyFile /etc/pki/tls/private/www.example.com.key (-private key)

SSLCertificateChainFile /etc/pki/tls/certs/example-ca.crt (-copy of all CA

certificates)

</VirtualHost>

ls -Zd /etc/pki/tls/

semanage fcontext -a -t cert_t "/etc/pki/tls(/.*)?" (-it is already the default)

restorecon -Rv /etc/pki/tls/

chmod 0600 /etc/pki/tls/private/*.key (-same as chmod u+rw *.key)

chmod 0644 /etc/pki/tls/certs/*.crt (-same as chmod u+rw,g+r,o+r *.crt)

e/ HSTS - strict transport security

<VirtualHost *:80>

Header always set Strict-Transport-Security "max_age=15768000"

RewriteEngine on

RewriteRule ^(/.*)\$ https://%{HTTP_POST}\$1 [redirect=301]

</VirtualHost>

f/ *Dynamic content*

I. *CGI*

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

ScriptAlias /cgi-bin/ "/var/www/cgi-bin/" (first parameter is part of the URL, second is the location of the script)

SELinux fcontext: httpd_sys_script_exec_t, httpd_enable_cgi

II. *PHP* (cp /etc/httpd/conf.d/php.conf ~/php.conf.orig)

```
yum -y install mod_php php php-mysql
```

```
<FilesMatch \.php$>
```

```
    SetHandler application/x-httpd-php
```

```
</FilesMatch>
```

```
    DirectoryIndex index.php
```

III. *Python*

```
yum -y install mod_wsgi
```

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

WSGIScriptAlias /myapp "/srv/my.py" (-a request for www.example.com/myapp will cause the server to run the WSGI application defined in /srv/my.py)

SELinux fcontext: httpd_sys_content_t

g/ SELinux: #man 8 httpd_selinux#

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

```
semanage port -a -t http_port_t -p tcp 88 (-for non-standard HTTP ports)
```

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

```
restorecon -Rv /srv/sitel/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

#14

SHELL ENVIRONMENT

a/ *Global*

```
/etc/profile
```

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

```
/etc/bashrc
```

b/ *User*

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

```
~/.bashrc
```

I. **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.

II. **RCs** are for running commands, setting aliases, defining functions and other settings that cannot be exported to sub-shells.

export MYVAR (-supplied MYVAR are marked for automatic export to the environment of subsequently executed commands)

alias

unalias

```
function () {...}
set
unset
```

#15

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:

a/ vim dbbackup; chmod +x dbbackup

#!/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 \$FMTOPTIONS -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

b/ 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\""

c/ 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

```

```
d/ vim 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%\""
```

```
e/ vim 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
                    #getent - get entries from Name Service Switch libraries, e.g. getent passwd
                    user, getent shadow user, getent ahosts|aliases|group|gshadow|hosts|networks|services...
                    echo "Invalid user"
                    exit
                fi
                getent passwd $2 | cut -d: -f7
                ;;
    *) exit
        ;;
esac
```