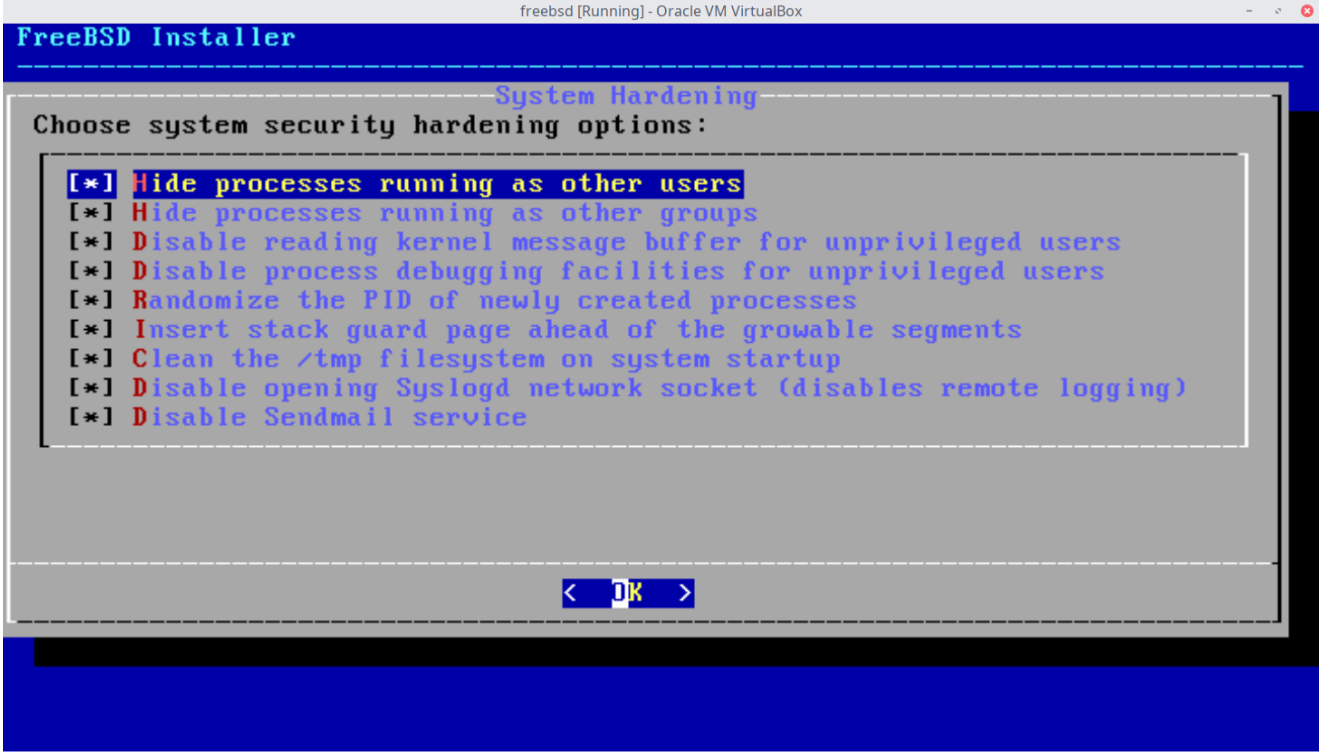
Staking Qtum on FreeBSD

FreeBSD is a very powerful operating system, it has a great history of reliability, security and stability. Here we show how it can be used to stake Qtum in a secure way.

On this tutorial, we'll make use of FreeBSD jails, here's a good read on Jails: https://www.freebsd.org/doc/handbook/jails.html

Installing FreeBSD

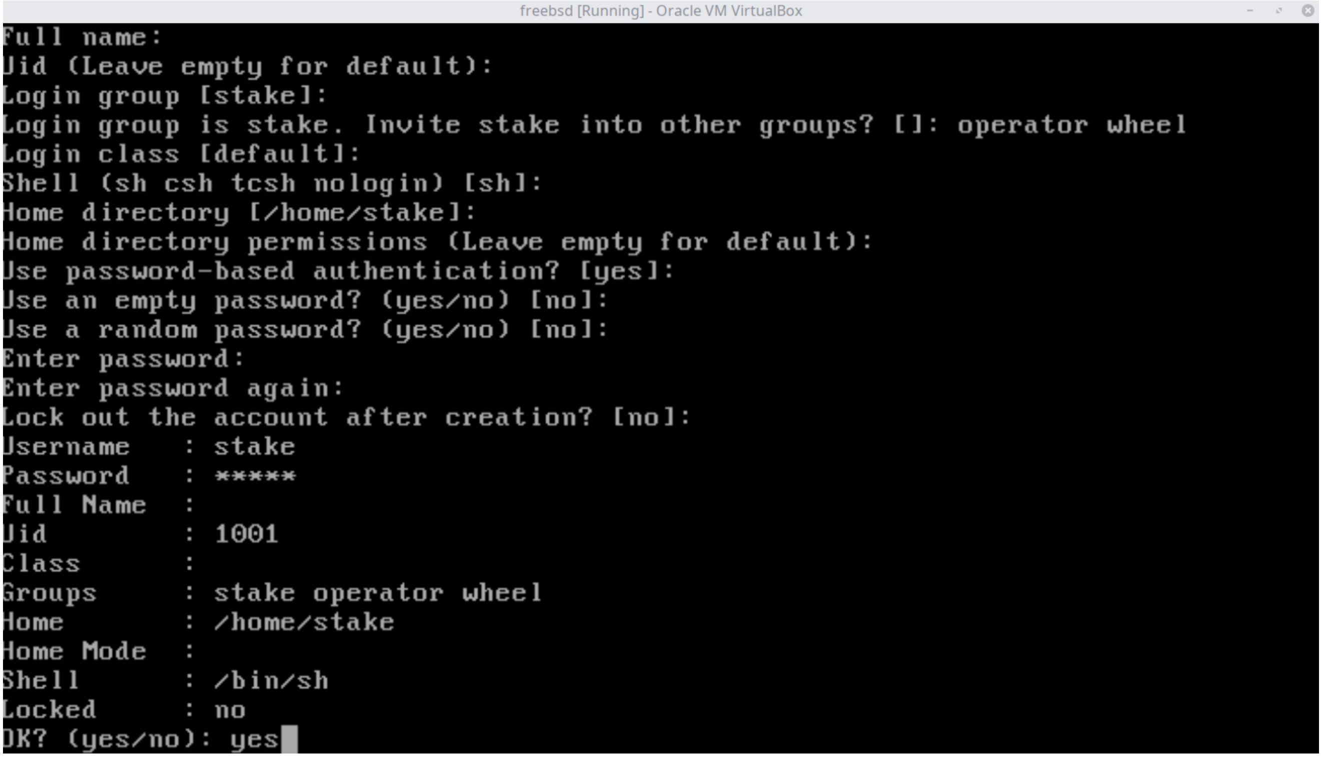
Install FreeBSD as normal, however, the following hardening settings are recommended during install:



```
Choose system security hardening options:

[*] Hide processes running as other users
[*] Hide processes running as other groups
[*] Disable reading kernel message buffer for unprivileged users
[*] Disable process debugging facilities for unprivileged users
[*] Randomize the PID of newly created processes
[*] Insert stack guard page ahead of the growable segments
[*] Clean the /tmp filesystem on system startup
[*] Disable opening Syslogd network socket (disables remote logging)
[*] Disable Sendmail service
```

Create user

Create a user with permissions "operator wheel"



```
Full name:
Uid (Leave empty for default):
Login group [stake]:
Login group is stake. Invite stake into other groups? []: operator wheel
Login class [default]:
Shell (sh csh tcsh nologin) [sh]:
Home directory [/home/stake]:
Home directory permissions (Leave empty for default):
Use password-based authentication? [yes]:
Use an empty password? (yes/no) [no]:
Use a random password? (yes/no) [no]:
Enter password:
Enter password again:
Lock out the account after creation? [no]:
Username    : stake
Password    : *****
Full Name   :
Uid         : 1001
Class       :
Groups      : stake operator wheel
Home        : /home/stake
Home Mode   :
Shell       : /bin/sh
Locked      : no
OK? (yes/no): yes
```

Please remember to do all these commands as root

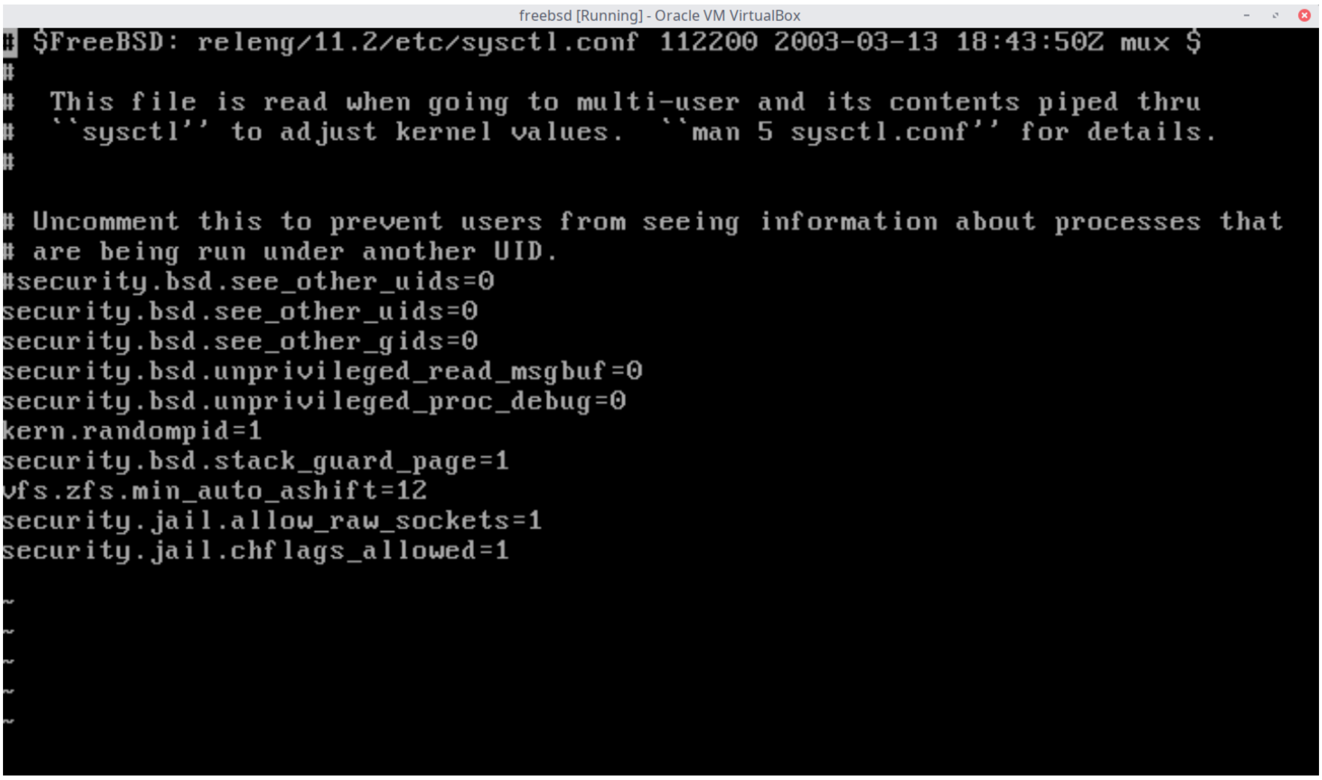
Host:

/etc/sysctl.conf

Allow sockets and upgrades in jail

`security.jail.allow_raw_sockets=1`

`security.jail.chflags_allowed=1`



```
# $FreeBSD: releng/11.2/etc/sysctl.conf 112200 2003-03-13 18:43:50Z mux $
#
#  This file is read when going to multi-user and its contents piped thru
#  ``sysctl'' to adjust kernel values.  ``man 5 sysctl.conf'' for details.
#

# Uncomment this to prevent users from seeing information about processes that
# are being run under another UID.
#security.bsd.see_other_uids=0
security.bsd.see_other_uids=0
security.bsd.see_other_gids=0
security.bsd.unprivileged_read_msgbuf=0
security.bsd.unprivileged_proc_debug=0
kern.randompid=1
security.bsd.stack_guard_page=1
vfs.zfs.min_auto_ashift=12
security.jail.allow_raw_sockets=1
security.jail.chflags_allowed=1

~
~
~
~
~
```

/etc/rc.conf

`firewall_enable="YES"`

`firewall_quiet="YES"`

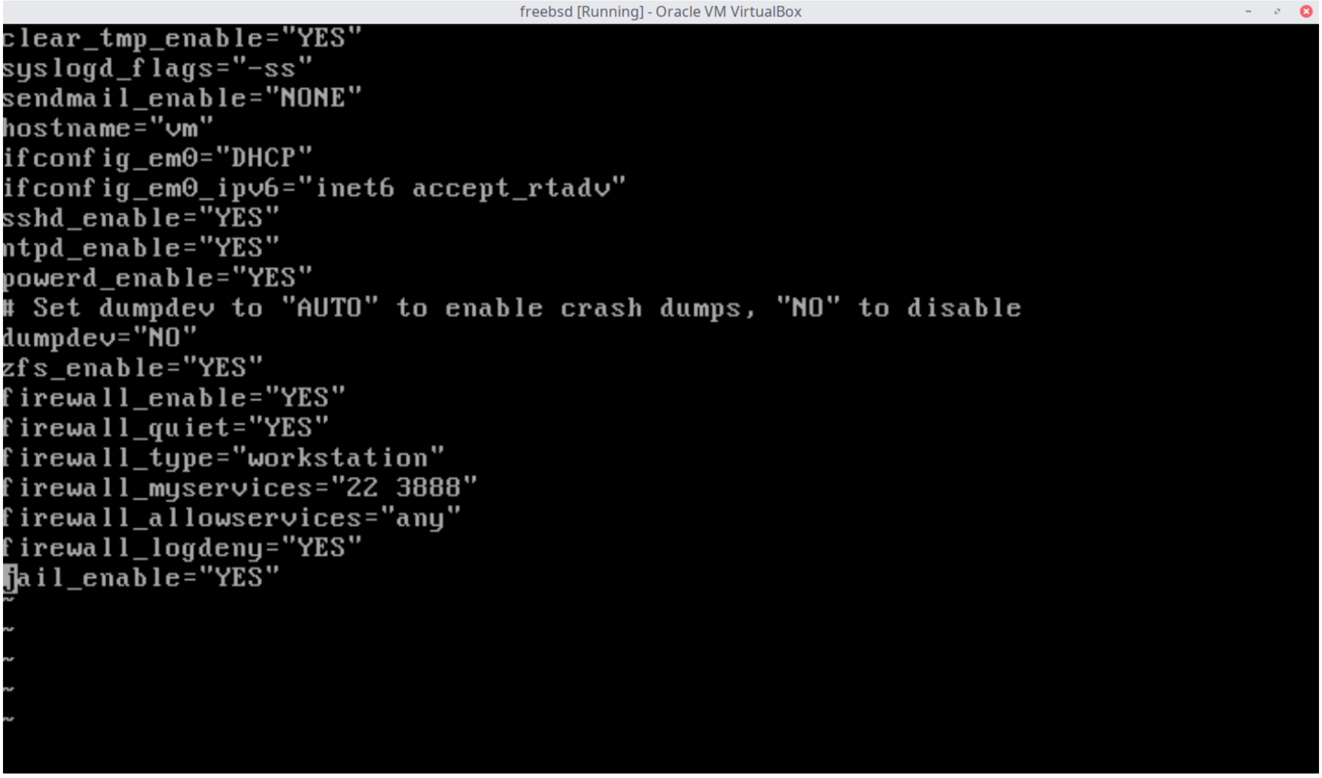
`firewall_type="workstation"`

`firewall_myservices="22 3888"`

`firewall_allowservices="any"`

`firewall_logdeny="YES"`

` jail_enable="YES"`



```
clear_tmp_enable="YES"
syslogd_flags="-ss"
sendmail_enable="NONE"
hostname="vm"
ifconfig_em0="DHCP"
ifconfig_em0_ipv6="inet6 accept_rtadv"
sshd_enable="YES"
ntpd_enable="YES"
powerd_enable="YES"
# Set dumpdev to "AUTO" to enable crash dumps, "NO" to disable
dumpdev="NO"
zfs_enable="YES"
firewall_enable="YES"
firewall_quiet="YES"
firewall_type="workstation"
firewall_myservices="22 3888"
firewall_allowservices="any"
firewall_logdeny="YES"
jail_enable="YES"
~
~
~
~
~
```

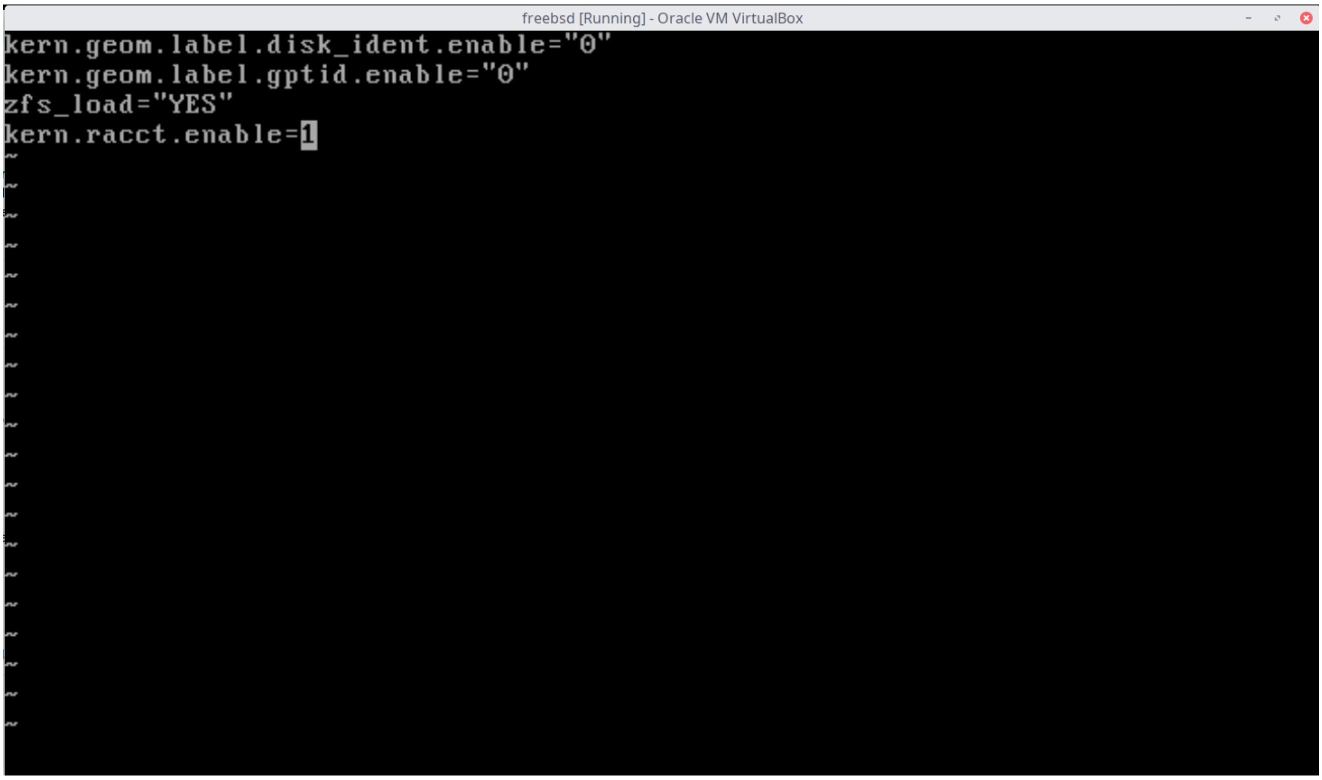
Notice that we've added some settings for firewall, these will enable IPFW and basic settings to secure our Jail, allowing only ports 22(ssh) and 3888(Qtum) to be accessed.

Resource limits for Jails

/boot/loader.conf

`kern.racct.enable=1`

` jail_enable="YES"`



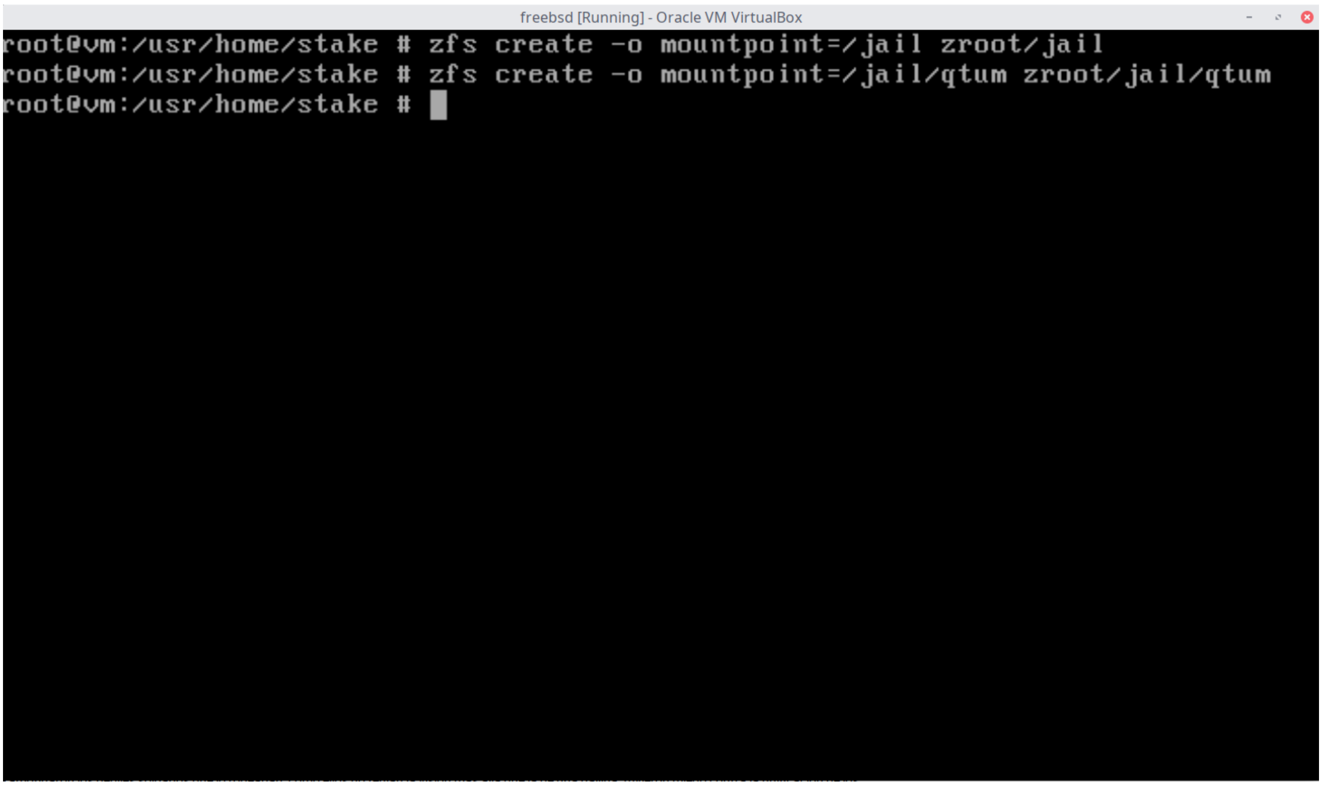
```
kern.geom.label.disk_ident.enable="0"
kern.geom.label.gptid.enable="0"
zfs_load="YES"
kern.racct.enable=1
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

Creating our Jail for staking

```
zfs create -o mountpoint=/jail  zroot/jail
```

(Change zroot for whatever name you chose for your zfs pool)

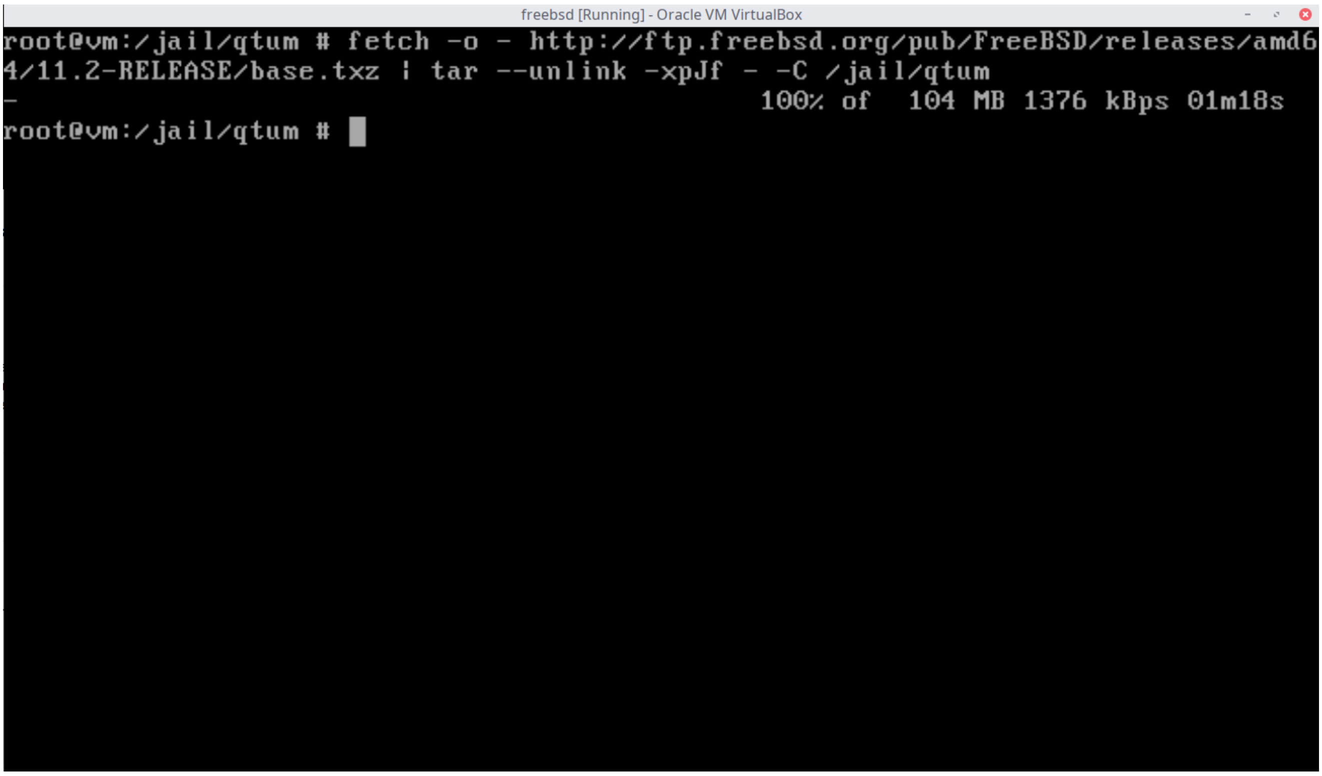
```
zfs create -o mountpoint=/jail/nextcloud  zroot/jail/qtum
```



```
root@vm:/usr/home/stake # zfs create -o mountpoint=/jail zroot/jail
root@vm:/usr/home/stake # zfs create -o mountpoint=/jail/qtum zroot/jail/qtum
root@vm:/usr/home/stake #
```

Now we've created our jail for staking Qtum, let's fetch and install FreeBSD on it!

```
cd /jail/qtum/ && fetch -o  -
http://ftp.freebsd.org/pub/FreeBSD/releases/amd64/11.2-RELEASE/base.txz | tar --
unlink -xpJf - -C /jail/qtum
```



```
root@vm:/jail/qtum # fetch -o - http://ftp.freebsd.org/pub/FreeBSD/releases/amd6
4/11.2-RELEASE/base.txz | tar --unlink -xpJf - -C /jail/qtum
-                                             100% of  104 MB 1376 kBps 01m18s
root@vm:/jail/qtum #
```

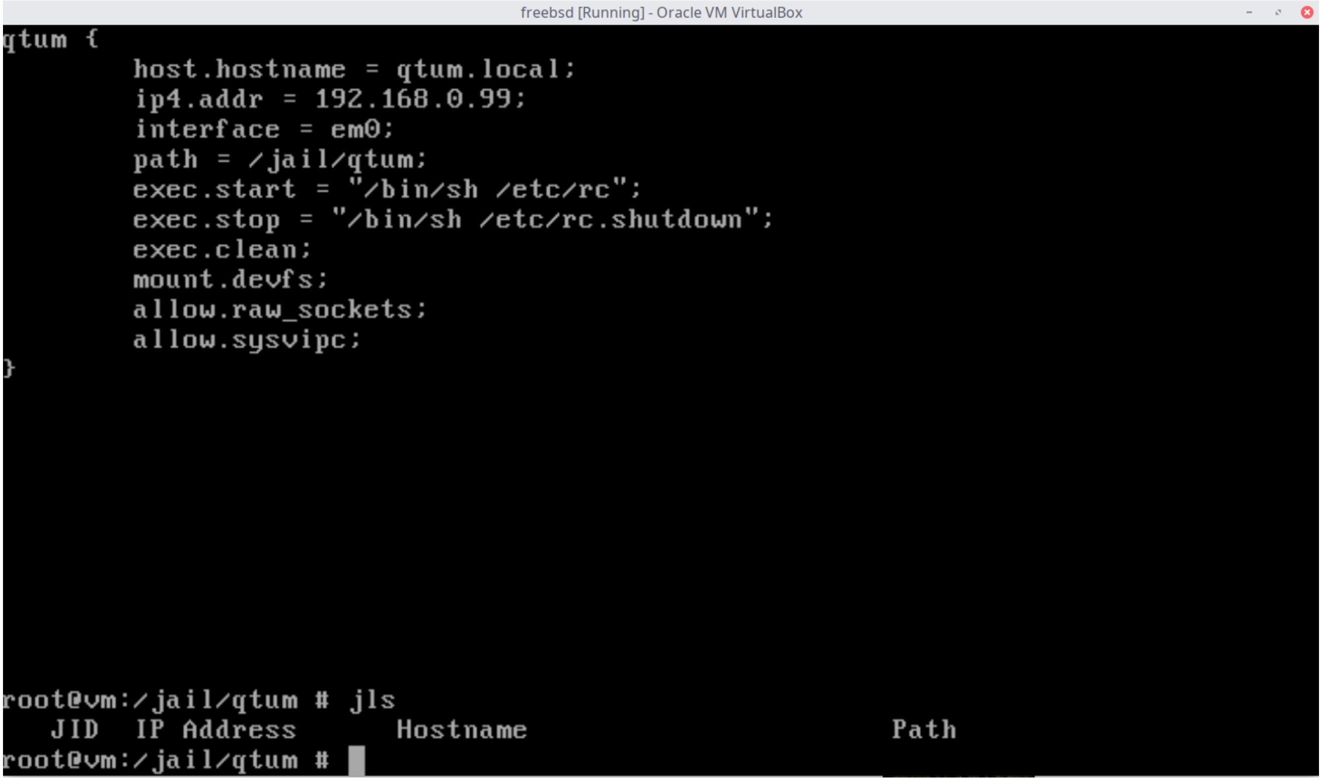
We've now installed FreeBSD into /jail/qtum

Typing ls /jail/qtum/ should show the filesystem of our Qtum FreeBSD Jail

Now, let's create the jail configuration file:

/etc/jail.conf

```
qtum {
  host.hostname = qtum.local;
  ip4.addr = 192.168.0.99;
  interface = em0;
  path = /jail/qtum;
  exec.start = "/bin/sh /etc/rc";
  exec.stop = "/bin/sh /etc/rc.shutdown";
  exec.clean;
  mount.devfs;
  allow.raw_sockets;
  allow.sysvipc;
}
```



```
qtum {
        host.hostname = qtum.local;
        ip4.addr = 192.168.0.99;
        interface = em0;
        path = /jail/qtum;
        exec.start = "/bin/sh /etc/rc";
        exec.stop = "/bin/sh /etc/rc.shutdown";
        exec.clean;
        mount.devfs;
        allow.raw_sockets;
        allow.sysvipc;
}
```

```
root@vm:/jail/qtum # jls
   JID  IP Address      Hostname                      Path
root@vm:/jail/qtum #
```

Ok now it's time to launch our jail!

```
service jail start qtum
```

We've just started our Qtum jail, We can now get into our Qtum jail to finish configuration, install Qtum and launch the wallet.

```
jexec qtum /bin/csh
```

```
cp /usr/share/zoneinfo/America/YOURTIMEZONE/ /etc/localtime
```

This is very important, if the time info is incorrect, we'll produce orphan blocks or will be unable to sync

Create our basic /etc/rc.conf for our Qtum Jail

Qtum Jail:

/etc/rc.conf

```
syslogd_flags="-s -s"
sshd_enable=YES
clear_tmp_enable=YES
clear_tmp_X=YES
extra_netfs_types=NFS
dumpdev=NO
update_motd=NO
keyrate=fast
```

`sendmail_enable=NONE`

`sendmail_submit_enable=NO`
`sendmail_outbound_enable=NO`
`sendmail_msp_queue_enable=NO`

Add dns nameservers to /etc/resolv.conf

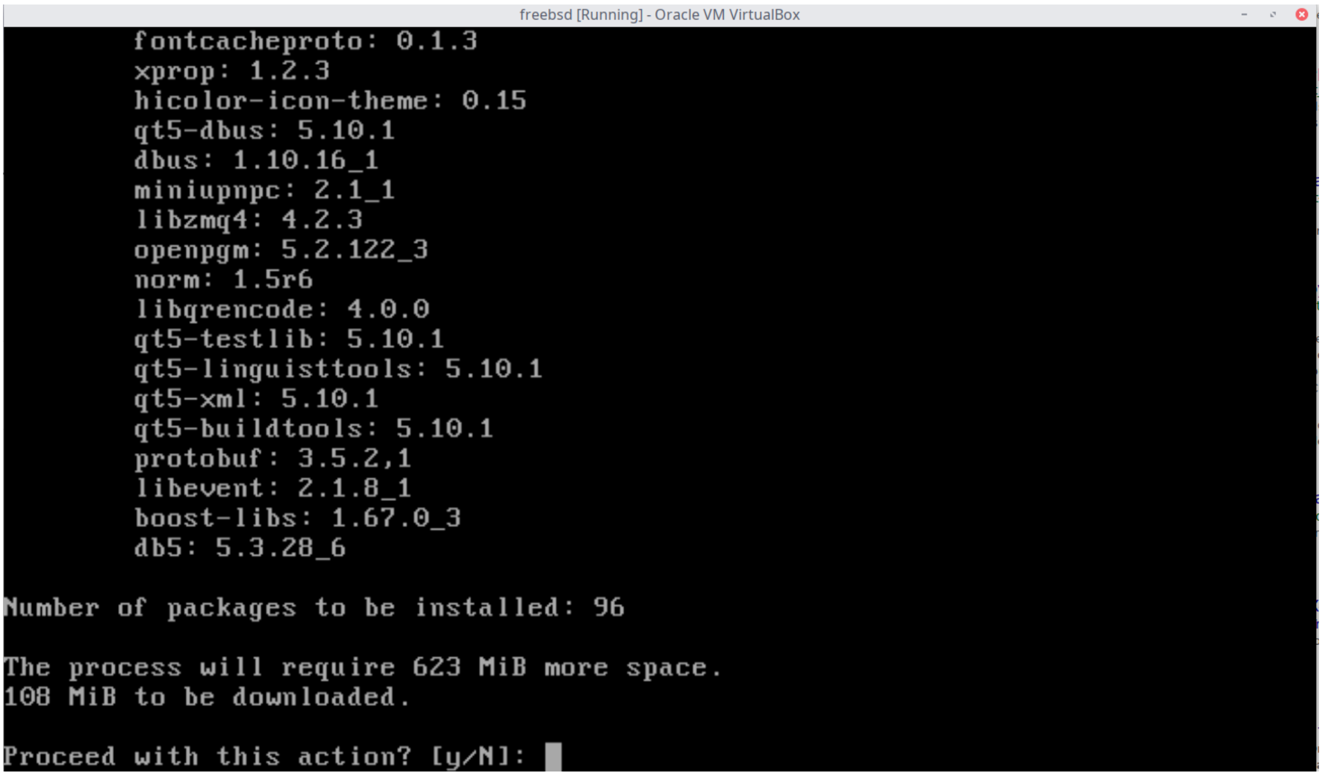
`echo "nameserver 8.8.8.8" >> /etc/resolv.conf`
`echo "nameserver 8.8.4.4" >> /etc/resolv.conf`

Installing Qtum

Now that we've got our jail up and running, we need to install Qtum.
There's 2 options on doing this, we can use the pkg repository or the powerful FreeBSD ports which are usually updated faster:

pkg repository

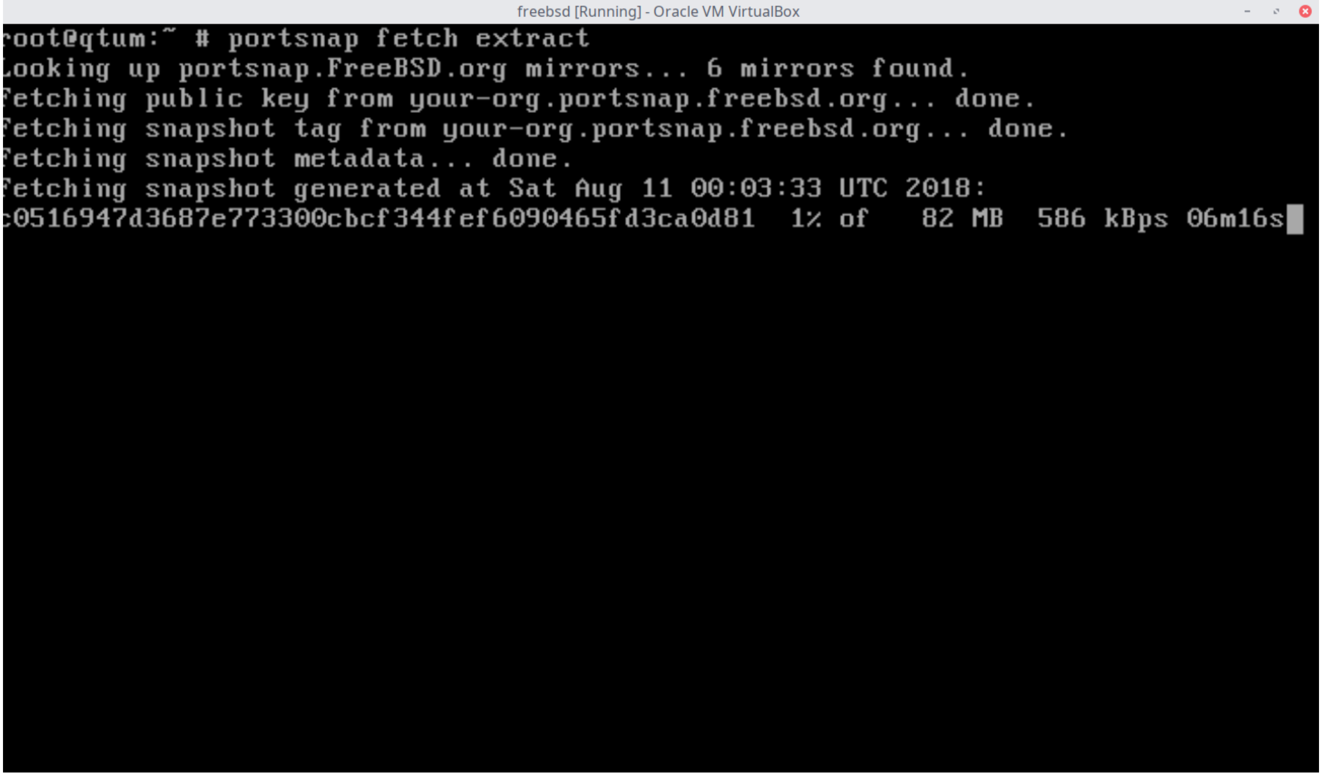
`pkg update -f`
`pkg install -y qtum`



```
        fontcacheproto: 0.1.3
        xprop: 1.2.3
        hicolor-icon-theme: 0.15
        qt5-dbus: 5.10.1
        dbus: 1.10.16_1
        miniupnpc: 2.1_1
        libzmq4: 4.2.3
        openpgm: 5.2.122_3
        norm: 1.5r6
        libqrencode: 4.0.0
        qt5-testlib: 5.10.1
        qt5-linguisttools: 5.10.1
        qt5-xml: 5.10.1
        qt5-buildtools: 5.10.1
        protobuf: 3.5.2,1
        libevent: 2.1.8_1
        boost-libs: 1.67.0_3
        db5: 5.3.28_6

Number of packages to be installed: 96

The process will require 623 MiB more space.
108 MiB to be downloaded.

Proceed with this action? [y/N]:
```

FreeBSD ports

`portsnap fetch extract`
`cd /usr/ports/net-p2p/qtum && make install clean`

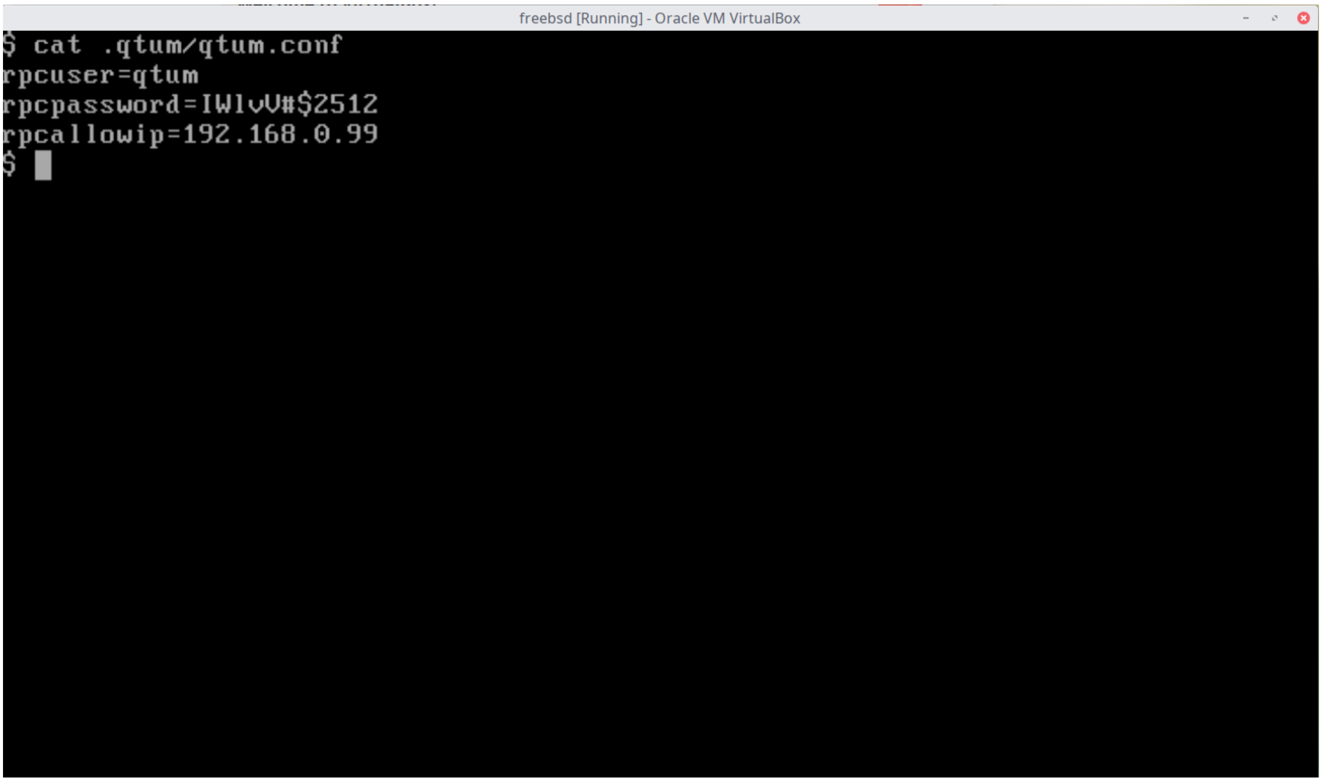


```
root@qtum:~ # portsnap fetch extract
Looking up portsnap.FreeBSD.org mirrors... 6 mirrors found.
Fetching public key from your-org.portsnap.freebsd.org... done.
Fetching snapshot tag from your-org.portsnap.freebsd.org... done.
Fetching snapshot metadata... done.
Fetching snapshot generated at Sat Aug 11 00:03:33 UTC 2018:
c0516947d3687e773300cbcf344fef6090465fd3ca0d81   1% of    82 MB  586 kBps 06m16s
```

The above will ask for a lot of configuration options, it might be better to use make config-recursive to set all options before compiling.
If you want to use default settings just type `cd /usr/ports/net-p2p/qtum && make install clean BATCH="YES"`

Running Qtum

Launching Qtum is just like in any other *NIX operating system, however there's a minor difference here due to how FreeBSD jails work. First, we need to create a qtum.conf file with the following contents:



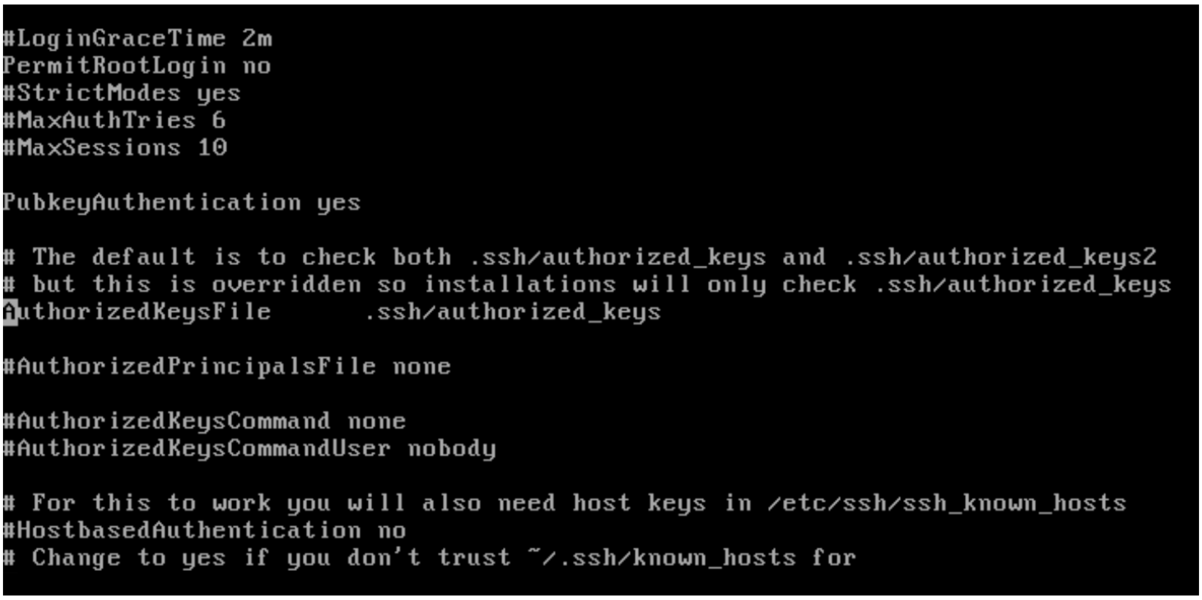
```
$ cat .qtum/qtum.conf
rpcuser=qtum
rpcpassword=IWlvV#$2512
rpcallowip=192.168.0.99
$
```

This config is necessary, otherwise calling the daemon will return errors.

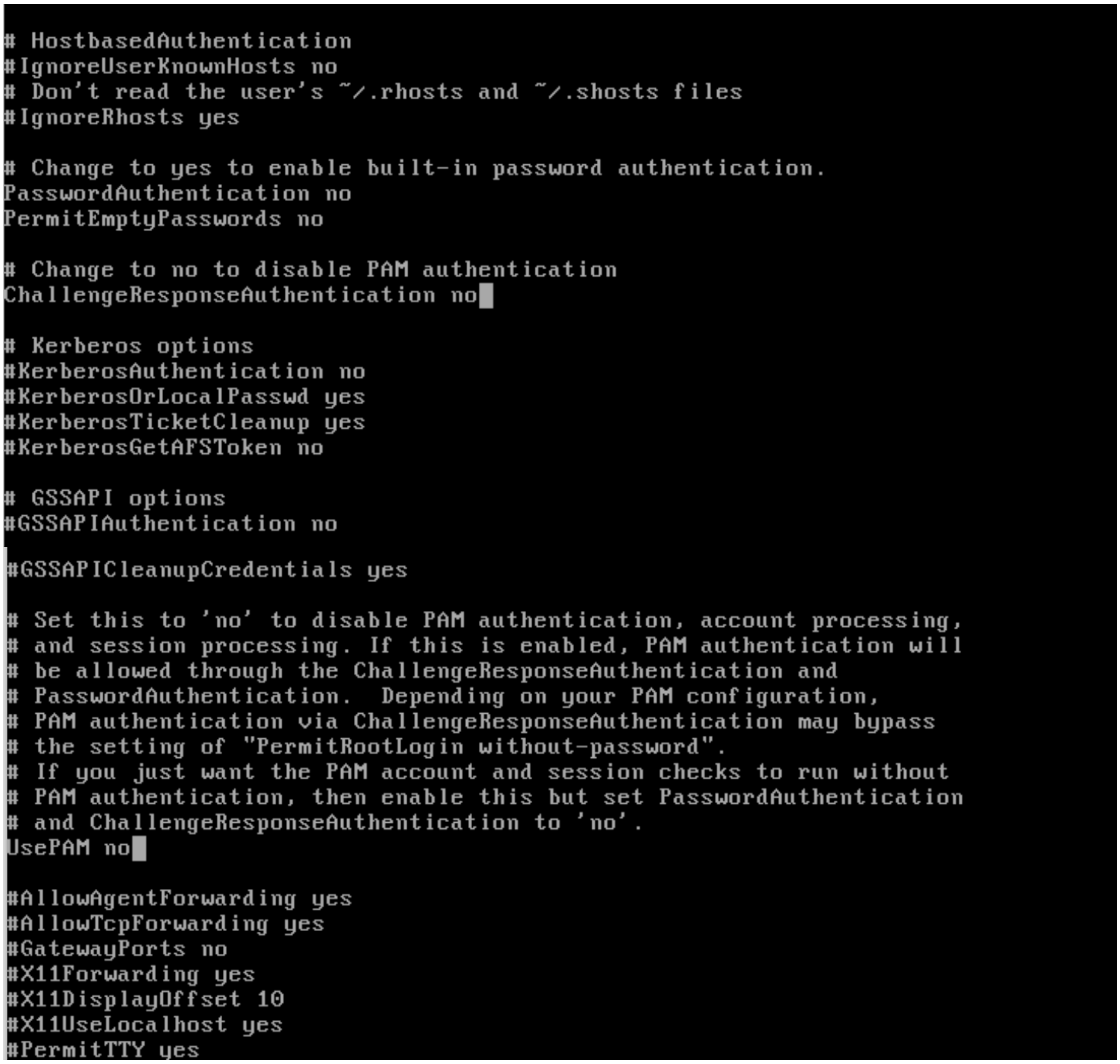
Then we can launch with `qtumd -daemon`

Security tips

- Set up firewall on host (you cannot setup a firewall inside a jail) and enable only the ports you need (22 and 3888)
- Disable history, this will completely disable console history and it's a way to help secure your staking box, type the following on your FreeBSD console: `unset history; unset savehist`
- Secure SSH:
  1. Disable password authentication

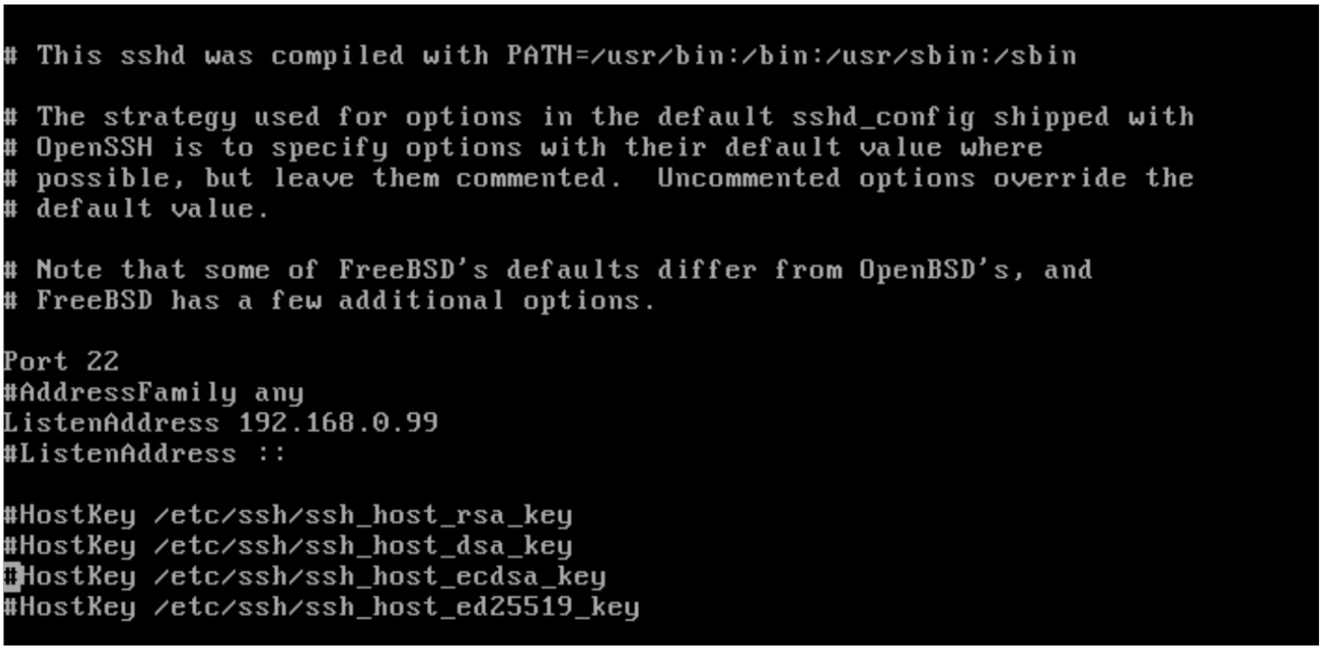


```
#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

PubkeyAuthentication yes

# The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
# but this is overridden so installations will only check .ssh/authorized_keys
AuthorizedKeysFile      .ssh/authorized_keys

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
```



```
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# Change to yes to enable built-in password authentication.
PasswordAuthentication no
PermitEmptyPasswords no

# Change to no to disable PAM authentication
ChallengeResponseAuthentication no

# Kerberos options
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes
#KerberosGetAFSToken no

# GSSAPI options
#GSSAPIAuthentication no

#GSSAPICleanupCredentials yes

# Set this to 'no' to disable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be allowed through the ChallengeResponseAuthentication and
# PasswordAuthentication.  Depending on your PAM configuration,
# PAM authentication via ChallengeResponseAuthentication may bypass
# the setting of "PermitRootLogin without-password".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and ChallengeResponseAuthentication to 'no'.
UsePAM no

#AllowAgentForwarding yes
#AllowTcpForwarding yes
#GatewayPorts no
#X11Forwarding yes
#X11DisplayOffset 10
#X11UseLocalhost yes
#PermitTTY yes
```

2. If using the FreeBSD box on your home network, force it to listen on local network only.



```
# This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

# Note that some of FreeBSD's defaults differ from OpenBSD's, and
# FreeBSD has a few additional options.

Port 22
#AddressFamily any
ListenAddress 192.168.0.99
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_dsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
```