Basic FreeBSD Installation

My basic FreeBSD installation is slightly different from the “norm” in that we manually configure the boot drive (usually an SSD) and add some additional packages we need. Here is the “Daily Data” way of building a server. Note that you have to be flexible; modify this as needed.

Installation

For the most part, we have a single disk for the boot drive. This will hold all of the software needed to have a working system. In order to make life easier, I remove all drives except the system drive, then boot from the FreeBSD installation thumbdrive.

When it comes time to partition the drive, I choose *manual* and set up from the command line. The main reason for this is that the FreeBSD installer in automatic mode does not allow you to set up without a swap partition, and I like **swap files** instead of swap partitions. While possibly slower, you have the flexibility to adjust the amount of space allocated to swap. This has saved me headaches in the past.

Also, just because it simplifies things, I remove all of the data drives. Normally, we have a single boot drive, then several drives which will contain data, generally as a ZFS file system. By removing the drives, I know which one is the one I want to install onto. This can cause problems since adding drives after a system is configured can rename existing drives. However, if your boot drive is on an internal connection (most modern servers have this capability) or you make sure it is in the first drive bay, drive renaming is not an issue.

From the command line, do the following. This assumes your boot drive is ada0. This is directly stolen from http://www.wonkity.com/~wblock/docs/html/ssd.html, though it is summarized here.

```
# create a gpt partitioning scheme on /dev/ada0
gpart create -s gpt ada0
# add a very, very small partition for boot
gpart add -t freebsd-boot -s 512k -a4k -l ssdboot ada0
# set it up to be bootable
gpart bootcode -b /boot/pmbr -p /boot/gptboot -i1 ada0
# now, add the rest of space as a second partion
# if you want, you can specify the size with the -s parameter
# as in '-s 100g' to only use 100G
gpart add -t freebsd-ufs -l ssdrootfs -b 1m ada0
# format the second partition.
# NOTE: the -t option below turns on TRIM for SSD's
# you can remove that if you are not using an SSD
newfs -U -t /dev/gpt/ssdrootfs
```

Note: the referenced article actually uses separate partitions for /var and /usr. In my case, we are generally setting things up in a way that this is just over complication, though in some installations it is necessary to break it down further.

Complete the installation. Be sure to add one user to the wheel group, or they will not be able to su to root. If you forget, when you log in for the first time (as root), add the user manually:

```
pw user  mod  username -G wheel
```

where *username* is the username you want to add.

Post Installation Partitions

Note that /tmp is missing and there is no swap space. The first thing I want to do is set /tmp and /var/tmp to use the same ramdisk (aka tmpfs). Assuming I have sufficient RAM, I can allocate some space for tmp, which makes things faster and cleaner.

Additionally, I want to create a swap file to replace the partition. Swap is very nice to have, but rarely used, but I had one case where my swap partition was just too damned small and the server started acting squirrely whenever there was a lot of ZFS activity.

```
# create a 4G file to be used for swap space. modify size as necessary
dd if=/dev/zero of=/swapfile bs=1G count=4
# Create the entry in fstab
echo 'md99  none  swap  sw,file=/swapfile 0 0' >> /etc/fstab
# create a tmpfs entry in fstab for /tmp
echo 'tmpfs  /tmp  tmpfs  rw,mode=01777 0 0' >> /etc/fstab
# move /var/tmp to point to /tmp
rm -fR /var/tmp
ln -s /tmp /var/tmp
# activate /tmp. This could cause instability
rm -fR /tmp/*
mount /tmp
# Just to be on the safe side, reboot
# might as well plug the rest of the drives in
# if you removed them as we normally do, then
reboot
```

Install some basic packages

I generally like some things that are not installed by default for FreeBSD (or Debian Linux, or Microsoft Windows, or Apple OSX, for that matter). For instance, I accept the larger size of bash for the extra functionality, and I'm lost without the *joe* editor. Some people are just more comfortable with a web UI than the standard CLI, so they might consider installing webmin (https://webmin.com). We will install

- joe (because its my favorite editor)
- postfix (because I hate sendmail)
- bash (a lot more robust than sh)
- perl5 (I write a lot of perl scripts)
- pv (very cool for long running copies)

- sudo (allows users to be elevated to root without giving them root's password)
- screen (very, very cool for long running processes)
- webmin (if you want a webui for managing a lot of things on the system)
- ipmitool (if this is a server with ipmi enabled functions)
- pbzip2 and xz (good compression technologies)

I'll label the steps as “webmin only” or “bash only” so you can easily not use them.

1. Install the packages - Answer 'Y' when asked if you want to enable postfix

```
pkg install joe perl5 pv pbzip2 xz sudo screen webmin ipmitool postfix
bash
```

2. Set up postfix and disable sendmail

```
service sendmail stop
sysrc postfix_enable="YES"
sysrc sendmail_enable="NONE"
mv /etc/mail/mailer.conf /etc/mail/mailer.conf.old
install -m 0644 /usr/local/share/postfix/mailer.conf.postfix
/etc/mail/mailer.conf
# clean up some leftover sendmail stuff
echo 'daily_clean_hoststat_enable="NO"' >> /etc/periodic.conf
echo 'daily_status_mail_rejects_enable="NO"' >> /etc/periodic.conf
echo 'Daily_status_include_submit_mailq="NO"' >> /etc/periodic.conf
echo 'daily_submit_queuerun="NO"' >> /etc/periodic.conf
# add postfix user to mail group so it has access to sasl
pw group mod mail -m postfix
# start postfix mail server
service postfix start
```

3. Set up bash

```
mount -t fdescfs fdesc /dev/fd
cp /etc/fstab /etc/fstab.bak
echo '# enable bash' >> /etc/fstab
echo 'fdesc  /dev/fd  fdescfs  rw   0   0' >> /etc/fstab
# you an now set the shell for any user with
chsh -s bash username
```

4. Set up webmin

```
/usr/local/lib/webmin/setup.sh
echo "webmin_enable="YES"" >> /etc/rc.conf
/usr/local/etc/rc.d/webmin start
```

5. Set up ipmitool

```
kldload ipmi
echo 'ipmi_load="YES"' >> /boot/loader.conf
```

References

- https://doxfer.webmin.com/Webmin/Installation
- http://www.wonkity.com/~wblock/docs/html/ssd.html
- https://www.cyberciti.biz/faq/freebsd-bash-installation/

From:
https://kb.unixservertech.com/ - **Unix Server Tech Knowledge Base**

Permanent link:
https://kb.unixservertech.com/unix/freebsd/system_builds/basic_freebsd_installation?rev=1572991695

Last update: **2019/11/05 22:08**