

MAGAZINE

BSD

FOR NOVICE AND ADVANCED USERS

DAY-TO-DAY BSD ADMINISTRATION

INSIDE

MIDNIGHTBSD CUSTOM INSTALLATIONS AND LIVE CDS

BSD SERVER MAINTENANCE

RE-PURPOSING AN ABANDONED MAC MINI
AS A WIRELESS ROUTER WITH OPENBSD

MONIT – MONITORING SOLUTION FOR ENTERPRISE
AND SOHO SERVERS WITH FREEBSD

A CLOSER LOOK AT THE CHANGES IN PC-BSD/TRUEOS 9.2

PART 2 – DIRECTORY ENCRYPTION USING PEFS

VOL.7 NO.9
ISSUE 09/2013(50)
1898-9144



855-GREP-4-IX
www.iXsystems.com
Enterprise Servers and Storage
for Open Source



- ✓ Rock-Solid Performance
- ✓ Professional In-House Support

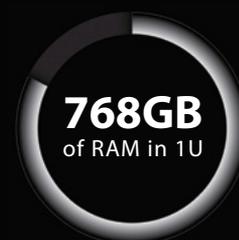
E5-2600

High Performance, High Density Servers for Data Center, Virtualization, & HPC



MODEL: iXR-22X4IB

<http://www.iXsystems.com/e5>



KEY FEATURES

iXR-22X4IB

- Dual Intel® Xeon® Processors E5-2600 Family per node
- Intel® C600 series chipset
- Four server nodes in 2U of rack space
- Up to 256GB main memory per server node
- One Mellanox® ConnectX QDR 40Gbp/s Infiniband w/QSFP Connector per node
- 12 SAS/SATA drive bays, 3 per node
- Hardware RAID via LSI2108 controller
- Shared 1620W redundant high-efficiency Platinum level (91%+) power supplies

iXR-1204+10G

- Dual Intel® Xeon® Processors E5-2600 Family
- Intel® C600 series chipset
- Intel® X540 Dual-Port 10 Gigabit Ethernet Controllers
- Up to 16 Cores and 32 process threads
- Up to 768GB main memory
- Four SAS/SATA drive bays
- Onboard SATA RAID 0, 1, 5, and 10
- 700W high-efficiency redundant power supply with FC and PMBus (80%+ Gold Certified)

Call iXsystems toll free or visit our website today! **1-855-GREP-4-IX** | www.iXsystems.com

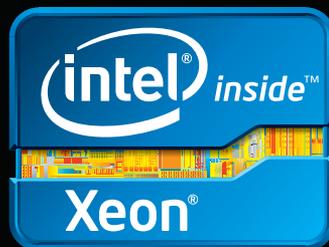
High-Density iXsystems Servers powered by the Intel® Xeon® Processor E5-2600 Family and Intel® C600 series chipset can pack up to 768GB of RAM into 1U of rack space or up to 8 processors - with up to 128 threads - in 2U.

On-board 10 Gigabit Ethernet and Infiniband for Greater Throughput in less Rack Space.

Servers from iXsystems based on the Intel® Xeon® Processor E5-2600 Family feature high-throughput connections on the motherboard, saving critical expansion space. The Intel® C600 Series chipset supports up to 384GB of RAM per processor, allowing performance in a single server to reach new heights. This ensures that you're not paying for more than you need to achieve the performance you want.

The iXR-1204 +10G features dual onboard 10GigE + dual onboard 1GigE network controllers, up to 768GB of RAM and dual Intel® Xeon® Processors E5-2600 Family, freeing up critical expansion card space for application-specific hardware. The uncompromised performance and flexibility of the iXR-1204 +10G makes it suitable for clustering, high-traffic webservers, virtualization, and cloud computing applications - anywhere you need the most resources available.

For even greater performance density, the iXR-22X4IB squeezes four server nodes into two units of rack space, each with dual Intel® Xeon® Processors E5-2600 Family, up to 256GB of RAM, and an on-board Mellanox® ConnectX QDR 40Gbp/s Infiniband w/QSFP Connector. The iXR-22X4IB is perfect for high-powered computing, virtualization, or business intelligence applications that require the computing power of the Intel® Xeon® Processor E5-2600 Family and the high throughput of Infiniband.



iXR-1204+10G: 10GbE On-Board



iXR-22X4IB

Intel, the Intel logo, and Xeon Inside are trademarks or registered trademarks of Intel Corporation in the U.S. and other countries.

MAGAZINE BSD

Dear Readers,

The September issue of BSD Magazine is dedicated to day-to-day BSD administration with the use of MidnightBSD custom installations and Live CDs, BSD server maintenance, directory encryption using PEFS and much more.

We start with Rob's column, where he discusses the future of Microsoft, the industry giant.

System administrators and hobbyists often wish to build custom install media with their own software packages, or live CDs for fixing systems. In the What's New section, Lucas Holt explores MidnightBSD custom installations and live CDs.

Many sites and handbooks explain how to install servers, but once a server is running, can an administrator keep it that way? The "BSD Server Maintenance" article by Devyn Collier Johnson will cover the basics of maintaining a BSD server.

Then you will have the chance to read Patrick Allen's article entitled "Re-purposing an Abandoned Mac Mini as a Wireless Router with OpenBSD", where he will try to bring abandoned hardware into the present with OpenBSD.

Next, by reading Antonio Francesco Gentile's article called "Monit – Monitoring Solution for Enterprise and SOHO Servers with FreeBSD", you will learn how to set up and manage a monitoring server based on monit.

In the Admin section, Rob Somerville presents the eighth part of our series on programming – "FreeBSD Programming Primer", where we will refine our jQuery menu and start building a user friendly interface to add content.

Then, Kris Moore talks a bit about Directory encryption using PEFS in the second part of his "A Closer Look at the Changes in PC-BSD/TrueOS 9.2" series.

After that, Carlos Antonio Neira Bustos talks about the Z file system in his "Intro to ZFS" article.

Finally, Angel Leon shows you the caveats of deploying FreeBSD over XenServer and the advantages over the traditional schema in improving administration, provisioning and delivery times.

We hope you will enjoy this issue and find many interesting articles!

Kamil Sobieraj
Editor of BSD Magazine
& BSD Team

Editor in Chief:

Ewa Dudzic
ewa.dudzic@software.com.pl

Supportive Editor:

Kamil Sobieraj
kamil.sobieraj@software.com.pl

Contributing:

Antonio Francesco Gentile, Devyn Collier Johnson, Rob Somerville, Patrick Allen, Antonio Francesco Gentile, Kris Moore, Carlos Antonio Neira Bustos, Angel Leon

Top Betatesters & Proofreaders:

Annie Zhang, Denise Ebery, Eric Geissinger, Luca Ferrari, Imad Soltani, Olaoluwa Omokanwaye, Radjis Mahangoe, Mani Kanth, Ben Milman

Special Thanks:

Annie Zhang
Denise Ebery

Art Director:

Ireneusz Pogroszewski

DTP:

Ireneusz Pogroszewski
ireneusz.pogroszewski@software.com.pl

Senior Consultant/Publisher:

Paweł Marciniak
pawel@software.com.pl

CEO:

Ewa Dudzic
ewa.dudzic@software.com.pl

Production Director:

Andrzej Kuca
andrzej.kuca@software.com.pl

Advertising Sales:

Kamil Sobieraj
kamil.sobieraj@software.com.pl

Publisher:

Software Media Sp. z o.o. SK
ul. Bokserska 1, 02-682 Warszawa
Poland

worldwide publishing
tel: 1 917 338 36 31
www.bsdmag.org

Software Media Sp z o.o. SK is looking for partners from all over the world. If you are interested in cooperation with us, please contact us via e-mail: editors@bsdmag.org.

All trade marks presented in the magazine were used only for informative purposes. All rights to trade marks presented in the magazine are reserved by the companies which own them.

Let's Talk

06 It's Lonely at the Top

By Rob Somerville

With the imminent departure of Steve Ballmer and Microsoft insiders adamant that Bill Gates will not return, what is the future for the industry giant?

What's New

08 MidnightBSD Custom Installations and Live CDs

By Lucas Holt

System administrators and hobbyists often wish to build custom install media with their own software packages, or live CDs for fixing systems. This flexibility can ease deployments in large environments or make it easy to bring your favorite OS along with you. Before customizing install media, it's important to look at the existing build system in MidnightBSD. There are two directories within the source tree for building media, release and nrelease. Release is used to build installation media, and nrelease (which originated from DragonFly BSD) is used to build Live CDs and USB flash media.

10 BSD Server Maintenance

By Devyn Collier Johnson

Many sites and handbooks explain how to install servers, but once a server is running, can an administrator keep it that way? This article will cover the basics of maintaining a BSD server.

18 Re-purposing an Abandoned Mac Mini as a Wireless Router with OpenBSD

By Patrick Allen

Don't let thousands of dollars worth of hardware that has been abandoned by Apple rot with old, unsupported software. Bring it into the present with OpenBSD.

22 Monit – Monitoring Solution for Enterprise and SOHO Servers with FreeBSD

By Antonio Francesco Gentile

The state of the network services offered by a company is the business card with which it presents itself to the world. Thanks to constant monitoring, it is possible to understand how and where to improve the infrastructure of the network in real time and detect any abnormalities.

Admin

28 FreeBSD Programming Primer – Part 8

By Rob Somerville

In the eighth part of our series on programming, we will refine our JQuery menu and start building a user friendly interface to add content.

34 A Closer Look at the Changes in PC-BSD/TrueOS 9.2 – Part 2 – Directory Encryption Using PEFS

By Kris Moore

Last month we took a look at how PC-BSD is implementing ZFS boot-environments, which can be a life-saver for both servers and desktops. This month we will be looking at how PC-BSD uses the PEFS kernel level file system module to automatically encrypt your home directory and its contents, and how you can manually run PEFS for other sensitive data.

38 Intro to ZFS

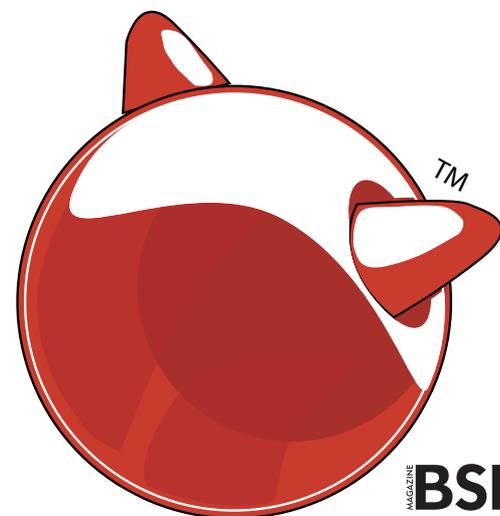
By Carlos Antonio Neira Bustos

“The Z file system, originally developed by Sun™, is designed to use a pooled storage method in that space is only used as it is needed for data storage. It is also designed for maximum data integrity, supporting data snapshots, multiple copies, and data checksums. It uses a software data replication model, known as RAID-Z. RAID-Z provides redundancy similar to hardware RAID, but is designed to prevent data write corruption and to overcome some of the limitations of hardware RAID.”

42 FreeBSD on XenServer

Angel Leon

In this article, we will learn the caveats of deploying FreeBSD over XenServer and the advantages over the traditional schema in improving administration, provisioning and delivery times.



It's Lonely at the Top

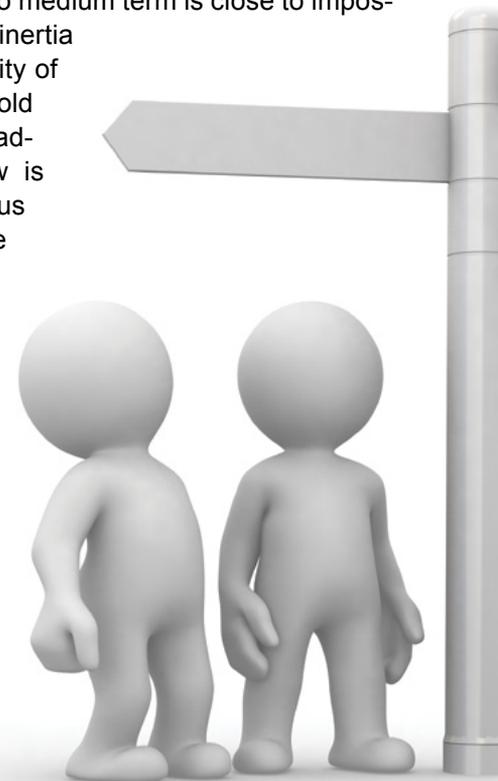
With the imminent departure of Steve Ballmer and Microsoft insiders adamant that Bill Gates will not return, what is the future for the industry giant?

A Chief Executive Officer was newly appointed to a large public owned organisation. On saying goodbye to his predecessor, he was handed 3 sequentially marked envelopes and advised to open each in turn when faced with a crisis. After an initial honeymoon period, sales were down so the first envelope was opened which said "Blame your predecessor". This worked for a while, until another major crisis arose, so the second envelope was opened which said "Reorganise". The CEO duly performed root and branch reform from remote site to corporate headquarters and this strategy seemed to work until another crisis emerged. On opening the third envelope, the CEO was greeted with the message "Prepare 3 envelopes".

While there is no suggestion that Steve Ballmer was pushed out of the role (he is retiring), the above anecdote illustrates well the challenges of being a CEO of a major technological blue chip. You can please some of the people some of the time, but rarely all at the same time. It seems that the Microsoft Corporation (NASDAQ: MSFT) is at a critical juncture both from the long term vision point of view and via strong competition in the sector. Looking back to Bill Gates' tenure, the organisation switched directions from being led by a technologist to being led by a marketeer. No doubt this made sense at the time, with the increasing move towards software becoming just "another consumer commodity". However, very much like Apple losing focus with the departure of Steve Jobs in 1985 and its subsequent rebirth upon his return, the strategy of running a technology company

by a non-technologist poses some interesting questions about corporate identity and its relationship to the marketplace. As the goal of any public corporation is to make as much money as possible for the shareholders, surely logic would dictate that a marketing/sales focused CEO would be the best candidate? Or maybe a banker or lawyer would be better. To the cynical, major organisations can be run by any discipline, as facilitating any effective change over the short to medium term is close to impossible due to cultural inertia and the sheer complexity of the business. The age old conundrum of what leadership model to follow is key; centralisation versus decentralisation? One large behemoth or lots of small divisions? Benevolent dictator or autonomy? Corporate clone or individual creativity? The permutations are endless, as history inevitability repeats itself under each new leader.

What is clear is that the original MSFT



vision of innovation is becoming decidedly threadbare. The pressure to drain every last profitable cent from the product is moving the market towards Software as a Service (SaaS). This might be fine for enterprises, but renting software will only remain a growing trend until the ever increasing pressure of vendor lock-in and the long term financial and security risk is realised by the customer. The big problem with renting anything (especially from an effective monopoly) is that they can easily dictate market price at a whim. At least if you own it (as far as you can own a software licence) you have a physical product until end-of-life. It is ironic that we have come full circle from the 1970's where enterprises rented processing on a time-share basis. This is a very different reality from building a company through innovation, aggressive acquisition and then branding the now ex-competitors' products as your own. The disconnect is even clearer as Bill Gates' vision of a computer on every desktop – a revolutionary idea at the time – has been such a clear success. MSFT is the dominant alpha male, effectively the industry standard whether in the enterprise or in the home. All of this is set to change with the rapidly growing mobile and tablet market, areas traditionally where MSFT has performed poorly. Time and again, the corporation has tried to gain an elusive foothold in alternative sectors without the obvious success it had in the 80's and 90's – Bing and Nokia being good examples.

In the past, MSFT managed to maintain pre-eminence with a simultaneous pincer attack on the enterprise and the consumer markets. This is now becoming much more asymmetrical, with MSFT holding its own in the corporate world and the consumer turning towards Apple and Android. The question is what will happen when XP finally turns end of life in 2014. Will the consumer stay

with Microsoft or decide to invest in alternatives? With XP holding an estimated market share of 40%, that is a lot of legacy software – and the cool enterprise reception to Windows 8 suggests Windows 7 will become the next consumer haven. With rumours of an aggressive development cycle for Windows 9, all this may change, but it is unthinkable that the user interface will change dramatically from the tablet-focused model. So in the

short term at least, MS has hedged their bets should their traditional domestic user base reject Windows 8 or 9 in a Vista-like fashion. Another variable is whether the

consumer market will choose to upgrade their desktop PC or move towards tablets. This further plays into the hands of other players (e.g. Google docs) and weakens another traditional cash cow – Microsoft Office.

Whoever leads up MSFT, it will be during a very turbulent and challenging time in the technology marketplace. If SaaS and the cloud continues its meteoric rise in popularity, the drift away from traditional licensing/purchase models will become a stampede. This is where MS faces a huge challenge – while it might be able to get away with charging the corporate customer, anyone can easily subscribe to free services like CloudOn, that offers MS Office on a tablet under a freemium pricing model – free to the majority, pay for premium services. Traditionally, the MS business model would be to purchase competitors and integrate, but how would they monetize this? Start charging and the user base will just move to Google docs unless they are totally locked into Office.

Microsoft needs to get back to its roots and decide what business model it wants to follow. As a technology company, they need a killer app, a widely adopted O/S across different platforms, and a revolutionary and exciting vision. As a service provider, they need to focus on the enterprise, especially in the area of Bring Your Own Device (BYOD) and solution/support provision. If they choose both, a fragmentation of the organisation seems inevitable, as the profit from traditional licensing models is evaporating and the business models are so vastly different. Could it be that MSFT is going the well trodden path of IBM, HP and Blackberry – too big and too slow to respond to a rapidly changing marketplace and consequently losing their onward inertia? Unless the new CEO can pull a rabbit out of the hat, the three envelope scenario looks increasingly likely sometime in the future. It's not only very lonely at the top, but very tough as well.

ROB SOMERVILLE

Rob Somerville has been passionate about technology since his early teens. A keen advocate of open systems since the mid eighties, he has worked in many corporate sectors including finance, automotive, airlines, government and media in a variety of roles from technical support, system administrator, developer, systems integrator and IT manager. He has moved on from CP/M and nixie tubes but keeps a soldering iron handy just in case.

MidnightBSD Custom Installations and Live CDs

System administrators and hobbyists often wish to build custom install media with their own software packages, or live CDs for fixing systems. This flexibility can ease deployments in large environments or make it easy to bring your favorite OS along with you.

What you will learn...

- how to create a live CD & flash media
- how to create custom install CDs

What you should know...

- how to build world, kernels and install packages using mport

Prior to customizing install media, it's important to look at the existing build system in MidnightBSD. There are two directories within the source tree for building media, `release` and `nrelease`. `release` is used to build installation media, and `nrelease` (which originated from DragonFly BSD) is used to build Live CDs and USB flash media.

Customizing Installation Media

Building a custom install image is helpful for system administrators when they need to replicate installations with the same configuration on multiple systems. The installation media can be given to technical support staff to install on desktops and servers throughout the organization. It also provides the benefit of a consistent install across machines, reducing maintenance headaches. In addition to customizing partitioning and packages, the sysadmin can build a custom kernel for the installation media with specific features enabled or disabled. Turning off unneeded features can improve security and performance. Another advantage with customizing install media is one can build the base system without specific features – such as games or services – for use on desktop systems. This limits what end users can do in a lab or other restricted computing environment. In server environments, it may be advised to disable Sendmail or BIND software when using other packages for mail and DNS services. Custom installation scripts can be added to install additional packages or do custom post configuration

tasks on installed services, similar to the standard `firstboot` script included in MidnightBSD's `rc.d`.

Why create a Live CD?

Live CDs and flash boot media can be useful to demo MidnightBSD to friends or to fix damaged installations. The MidnightBSD release CD in 0.4 has a simple live CD repair mode, but it has a limited set of tools available. By building a custom CD, one can include tools such as `fusefs` for working with extra file systems, partitioning software beyond `gpart`'s functionality or other custom packages for network security. Antivirus software could be used to scan infected machines on the network. Network scanners and other network security tools could be used to review network security from a trusted portable environment. It is also possible to create a custom live CD that imitates a full desktop environment and then create custom installation media to copy the entire environment to a local hard drive after testing on the system. Live CDs can even be used to test MidnightBSD on laptops at stores prior to purchase to ensure they're going to work, including testing X.org on the video card and verifying sound and networking are detected.

Creating a Live CD

Creating a CD requires building the world and kernel, and then using DragonFly's `nrelease` architecture. After running

the `realquickrel` target, an ISO file will be located at `/usr/release/mbsd.iso`.

Listing 1. Create a Live CD

```
# Build a Live CD on 0.3 or 0.4, assumes CVS checkout
cd /usr/src && make buildworld buildkernel
cd /usr/src/nrelease && make realquickrel
```

Customizing Live CDs

To customize a live CD, modify the contents of `/usr/src/nrelease/root`. Make any relevant configuration changes to etc files. These are copied into the staged directory.

Install packages from `mports` or using packages from the FTP server into the destination environment. This procedure assumes one is using MidnightBSD 0.4-RELEASE.

- `cd /usr/src; make buildworld`
- `make buildkernel`
- `cd nrelease`
- `make buildiso`
- `mkdir /usr/release/root/usr/mports`
- `cd /usr/release/root/usr; mount_nullfs /usr/mports mports; cd ..`
- `mport install <package name> # repeat as necessary for each package to install`
- `umount /usr/mports`
- `cd /usr/src/nrelease; make buildiso2`
- `make mkiso`

If you wish to use locally generated packages rather than files from the MidnightBSD FTP, use `/usr/libexec/mport.install` to install local packages. This will require a full path to the package.

Creating a USB Flash Environment

Creating a flash image for MidnightBSD 0.4 can be done in a similar manner to the Live CD or using the new `memstick` target in `src/release/Makefile`. One could customize the contents of `/usr/rel` prior to using the `memstick` target to create custom media. For further examples of creating a flash drive using the older Live CD method, consult the MidnightBSD wiki.

Listing 2. Copying the Flash Image to Media

```
# (assumes flash is /dev/da0)
dd if=${imgoutfile} of=/dev/da0 bs=1m
```

Customizing Installation Media

To build MidnightBSD releases, start by setting up a fresh machine with the same version of the OS. Normally, we

On the Web

- <http://www.midnightbsd.org/> – MidnightBSD Project website
- <http://www.midnightbsd.org/wiki/FlashDriveBoot> – MidnightBSD wiki on Flash Boot.

pull it from source, run the regular `buildworld`, `buildkernel`, `installkernel`, `reboot`, `installworld`, and `mergemaster` steps. Assuming you have already run `buildworld` and `buildkernel`, you can `cd /usr/src/release`. From the release directory, you will find a `Makefile` with several targets and environment settings.

Bsdinstall is the future

With 0.4-RELEASE, we adopted FreeBSD's `bsdinstall` based installation mechanism. Rather than place all the logic in one massive program like `sysinstall`, the installer is a series of simple programs and scripts. This creates flexibility in adding new install steps to the process and makes the `firstboot` script unnecessary.

Including Packages

As 0.4 includes a Live CD like workflow, it would be possible to generate an `index.db` file for the packages one wishes to include on the CD, place it in the installation media in `/var/db/mport` and include packages on the installation media. Then, install the packages using a script with calls to `/usr/libexec/mport.install`. The only caveat to this approach is that `mport` is designed to check automatically for updated index files on some operations. `mport.install` will not do this. The SQLite `index.db` file contains the URLs for mirror sites and could be modified to use local internal servers. It might be wise to modify the bootstrap URL in the `libmport` source code to point to your own mirror so a client does not fetch an index from the main MidnightBSD mirror for private installations of this type.

Summary

Building custom installation media can be useful for deploying MidnightBSD on a large number of systems in the same manner. Live CDs and flash media allow one to bring a custom MidnightBSD install with them or create rescue media for recovering damaged systems.

Using the makefiles in `nrelease` or `release` directories within a checked out copy of MidnightBSD sources allow you to build custom media.

LUCAS HOLT

Lucas Holt is the founder of the MidnightBSD project and a Senior Application Programmer/Analyst for the University of Michigan in Ann Arbor, MI, USA.

BSD Server Maintenance

Many sites and handbooks explain how to install servers, but once a server is running, can an administrator keep it that way? This article will cover the basics of maintaining a BSD server.

What you will learn...

- How to maintain any BSD server.
- Some basic rules of server maintenance.
- Some helpful shell commands.
- The crucial importance of logs.
- Backing up BSD servers.
- Security.
- Finding problems.
- Some basic information that will help prep admins for their BSD and/or LPI certifications

What you should know...

- Basic shell commands.
 - Basic network concepts.
 - Understanding of BSD and Unix systems.
 - General idea of the different server services and daemons.
-

Many server administrators may understand how to set up a FreeBSD server, but once it is up, can they keep it running? Many companies may store business transactions on servers like stock lists and item prices. If the server(s) goes down, a part of or the entire company is unable to continue or perform well until the server is back up. Also, if the server is used to host a website that sells products, customers cannot make purchases until the server is fixed. This means that the company loses money until the server is back online. If the data is lost, this means the company loses more profits. Clearly, it is important to keep the servers healthy and the data safe and secure.

When dealing with servers, the two most important directories that all administrators should thoroughly understand and learn are `/etc/` and `/var/`.

NOTE

I am using FreeBSD v9.1 for my descriptions and examples, but this article is valid for any BSD distro and many Linux/Unix operating systems.

NOTE

This article will not discuss how to install a server. It will only explain server maintenance and protection.

Logs

When maintaining a server, it is best to prevent issues before problems arise. Thankfully, many servers have a log system. Most applications/services have a log that lists the software's status and errors. When an error is spotted, the administrator should take the time to investigate and prevent future disasters.

Logs are stored in `/var/`. The logs are given names that easily identify their content or purpose. For example, the Samba log would be `/var/log/smb.log`. Some applications (rarely) store logs in the current user's home folder. If a log cannot be found, check in Root's home folder or read the application's manual for the log location.

If the HTTP services are having specific issues and the IT tech does not want to spend a lot of time searching through the Apache log, then the tech can do a specific search like below.

“ IN SOME CASES
nipper studio
HAS VIRTUALLY
REMOVED
the **NEED FOR** a
MANUAL AUDIT ”
CISCO SYSTEMS INC.

Titania's award winning Nipper Studio configuration auditing tool is helping security consultants and end-user organizations worldwide improve their network security. Its reports are more detailed than those typically produced by scanners, enabling you to maintain a higher level of vulnerability analysis in the intervals between penetration tests.

Now used in over 45 countries, Nipper Studio provides a thorough, fast & cost effective way to securely audit over 100 different types of network device. The NSA, FBI, DoD & U.S. Treasury already use it, so why not try it for free at www.titania.com



```
grep SOME_ISSUE_TO_SEARCH /var/log/httpd-error.log
```

For example, when searching for a connection error, search for words like “connection”, “failed”, “404” (HTTP connection error number), and other similar terms.

A server maintainer may use a command like the above to search the boot-up logs for lines containing the word “Ethernet” when there appears to be an Ethernet issue. There is a command specifically for viewing the boot-up log seen below. The command below displays the contents of `/var/run/dmesg.boot`.

```
dmesg -a | less
```

This command will allow the administrator to scroll through the boot messages. When searching for entries concerning the USB system, for instance, try the below command.

```
dmesg -a | grep usb
```

It is important to occasionally browse the output of the `dmesg` command, because sometimes errors may start occurring with the hardware or boot-up process. If the server has problems booting up, the network's or company's performance may suffer.

Paper Logs

It is best to keep a physical log in the server room or some other secure storage location. This log should include a list of all files changed, software installed, and hardware repaired along with answers to the following questions – who, what, why, when, where, and how.

Who

It helps to keep a written record of all IT techs that have dealt with the server. This helps keep track of all who may be aware of an error. Assume the server is having errors with the NTP services again. If Bill, the IT tech, was the last one to fix it, it may be best to have Bill check the issue again since he is familiar with it or to ask Bill for suggestions. If the server always tends to break after Andrew makes changes to the server, it may be best to investigate the issue.

What

IT techs may find it helpful to keep track of what other techs are fixing. If there is a network connection issue and the Ethernet devices have already been replaced a few times, then it may be safe to assume the problem lies somewhere else. Also, if the reason techs are checking

the server is because of reports of Samba issues, then the Samba services probably need to be completely re-installed (remove `/usr/local/samba` and run `pkg_add -r samba35`).

Why

Knowing why a particular software or hardware is being updated, repaired, and so on, may help narrow down future issue origins when finding the source of an error. For illustration, if Bill reinstalled the NIS services because they repeatedly locked up, then Casey knows that another reinstallation will not fix the same issue that is occurring the next day.

When

The time errors occur or when they were fixed can assist some techs. For example, if the server is always powered up at 5:00am and `syslogd` crashes at 5:03 am everyday, then most likely a service is starting up that conflicts with `syslogd`. This would then point to the cron tables, at tables, or `/etc/rc*` scripts (some BSD systems structure their rc system differently). Any process that starts at 5:03am or a little before may be causing the crash. As another example, if the last backup was performed a week earlier, then the tech looking over the paper log should perform a backup or assign the task to someone else. This time log would also keep administrators from ruling out causes that are indeed the cause. For instance, assume the server has a virus. The virus scanner is not finding the virus for some unknown reason. If the tech sees that the last anti-virus update was three months ago, then the tech has a good assumption that updating the anti-virus definitions will enable the scanner to find and remove the virus. Otherwise, the tech will falsely assume the scanner is up-to-date and waste precious time looking elsewhere for the fix for the problem.

Where

This may not be a very useful piece of information, but it may help in some instances. This would answer “from where was the server managed/fixed?” Servers can be fixed or repaired locally, remotely via secure shell (SSH), or other methods. If there are suspicions that the server is being hacked from a certain IP address, but it is later seen that the log states Andrew proof-read Bill's `/etc/rc.conf` edits through a remote shell using the suspicious IP address, then we can rule out that address.

How

This is an important piece of information. How was the problem fixed and diagnosed? Was the DNS server's

hard-drive replaced or reformatted? What command was used? This helps future problems get repaired the right way (if it worked) or differently if the issue still persists. This also helps to inform other techs what has already been done and tried with the server.

Also, when making such logs, include which server (if there are more than one) and which OS (if the server is set up for a dual-boot or virtual system). It may help to have a separate file for each server. A log containing errors for an unsolvable issue should be printed out for future reference just in case the server loses its files. It will be more difficult to fix an issue if the logs cannot be viewed.

Finding Problems

If users report that they cannot access files or information, then it may be wise to rule out as many issues as possible.

On some of the workstations, type `ping <SERVER_IP_ADDRESS>`. If the ping fails, then this proves the problem lies in the network connection. To help better pin-point the problem area, type `traceroute <SERVER_IP_ADDRESS>`. This command will print each location between the workstation and server. If the connection stops at a router, then the router or the devices before or after the router may be the cause of the connection error. If the connection is fine, the last stop should be the server.

Once it has been proven that the network is fine, it may mean the server itself is the source of the problem. If the inaccessible files are accessed via FTP, then the administrator may want to make sure the FTP process is still running. In a command-line, type:

```
pgrep ftpd
```

If the FTP daemon is running, the administrator should get a number (the PID) as the output. Then, the tech may want to restart the daemon using the command below.

```
service ftpd restart
```

If no output is received, then the FTP process closed for some reason. To turn it back on, type:

```
service ftpd start
```

The FTP server should be back online. Now would be a good time to investigate what caused the FTP daemon to close. To do this, check the logs. The log file for ftpd is `/var/logs/ftpd`. This is a plain text file. In a command-line, it can be read in one of many ways. The best way is to use the `less` command which allows the user

to scroll up and down. To see the last ten lines, use the `tail` command.

```
less /var/logs/ftpd
tail -10 /var/logs/ftpd
```

If the FTP daemon is not starting when the server turns on, then the administrator needs to check `/etc/rc.conf` and make sure this line is in the configuration file `ftpd_enable="YES"`. If not, then that is why the daemon never started; no script told it to start.

If for some reason the above still does not help, check some of the permission files. Administrators can set up which users are allowed or not allowed to access certain services. The file `/etc/ftpusers` lists users that are not permitted to access any service or file provided by the FTP daemon. If many or all users are listed here, then that would explain why no one can access the files. To allow everyone to use the FTP services, type:

NOTE: Remember to keep a backup of the `ftpusers` file by copying it to the root's home folder or some other designated backup location.

```
echo "" > /etc/ftpusers
```

This will erase the list and allow all users to use the FTP server. If there are some users that should not access the FTP portion of the server, make a list of the users and re-add them to the list using a preferred text editor. Many users use Vi or Emacs in the command-line.

To make sure the BSD server is recognizing all of its network devices, type `ifconfig` and make sure all of the Ethernet ports and wireless devices are listed. If any of the network devices are missing, then the operating system is missing a driver or that device is physically broken.

Checking the File System

If the filesystem goes bad, then the contained data will be damaged and lost. Performing a simple restoration from a backup cannot be done. Look at the filesystem as a landscape or garden. If the soil becomes rocky and bad for the plants, a gardener cannot replant a new plant without making the soil healthy. The same goes for a filesystem. To check a filesystem for errors, use `fsck`.

```
fsck -F ufs
```

The command will check the UFS filesystem for errors. If an error is found, it may be wise to make an immediate backup of important savable files and then reformat or replace the hard-drive. Remember, when checking a

filesystem using `fsck`, specify the filesystem type to be inspected. Otherwise, the check will not work properly if `fsck` is checking a UFS filesystem while expecting it to be ZFS.

Generally, the best way to prevent or ease the repair of bad filesystems is to use a RAID system with parity. Then, the damaged storage unit can be removed and replaced with a new or repaired unit. FreeBSD will then recreate the data using the parity system.

Problems with Finding Problems

Logs are very useful in solving many problems. However, logs may not always be there to help. For instance, if the server locks up, `syslogd` (the process that makes logs) will not be able to write the logs. Then when the logs are viewed, nothing will be seen for the sequence of events that led up to the disaster.

If malware erases the hard-drive, no logs will be seen. Also, if the hard-drive or filesystem is corrupted, then no logs will be seen either.

GUI

The default FreeBSD installation lacks a graphical user interface (GUI). For a server, the administrator must take some details into consideration before installing a GUI. A GUI would make a system easier to repair and maintain. However, this would make it easier for someone within the company to ruin the server and its data. Also, there would be more software that could cause a conflict with existing programs.

If for some reason a GUI must be installed on a server, it is best to install a graphical user interface with a small footprint (uses very little resources). Clearly, KDE, GNOME, Mate, Cinnamon, and Unity are not good choices for desktop environments on a server. Some graphical interfaces suitable for a server include Afterstep, Ratpoison, Enlightenment, Blackbox, Fluxbox, and other similar interfaces. XFCE or LXDE may work well on a server, but it may be best not to install a desktop interface that large on a server.

NOTE

At the time of writing, BSD distros have recently started using graphical user interfaces. However, they must be downloaded, compiled, and installed. BSD systems may have problems with some graphical user interfaces. It is best to avoid desktop interfaces unless it is absolutely necessary.

Quick Fixes

If a server daemon is found to not run on boot-up, of course, it needs to be added to `/etc/rc.conf`. However,

some administrators may not like Vi or Emacs and do not have time to install a preferred text editor. Well, there is a quick fix. To quickly and easily make a daemon start when the system loads up, type the following command:

```
echo "apache22_enable=\"YES\"" >> /etc/rc.conf
```

If an entry in `/etc/rc.conf` is spelled wrong, it can swiftly be corrected. Assume the above command added the misspelled line `apachy22_enabled="YES"`. Type the below command to make the correction.

```
cat /etc/rc.conf | sed -i -e 's|apachy22|apache22|'
```

The above command will perform a find-and-replace in file (changes take place instantly). No regex (wildcards) is used, so the exact string will be matched and changed. Unless the administrator is very skilled with regex and has thoroughly read the configuration file, no one should use regex in such an important system file. Otherwise, settings that should be left alone will get changed if the tech is not careful. This can cause the server problems, and the administrator will have to spend time finding and fixing the problem.

Updating

Updates can be good for a server, but they can also be harmful. Updates may offer bug fixes, new abilities, more efficient algorithms, and less resource usage. So, updates may help a server's performance. However, an update may contain a bug that the developers did not find. This bug may be minor or it could cause the system to be down for a while. Generally, it is best to have a "testing server". This server would be exactly like the main server, but the testing server has the latest updates. Server administrators would use such a server to test out new configuration settings on services and make sure that updates and new software work properly. However, some companies may not have the funds to have this testing system. It may also help to watch the Internet for reports on major bugs.

An alternative to a testing server is to have a virtual testing server. Install visualization software on a computer/operating-system of choice. Then, install BSD and test the newest updates and such on this system. This will not be a perfect test because the hardware is not the same as the server. BSD distros run very well in virtual machines, so no problem should exist here. When updating the system, type:

```
freebsd-update fetch; freebsd-update install
```

This will update the list of available software and then update the software. You must be root to apply such updates.



vBSDCon • SAVE THE DATE!

DULLES, VA • OCTOBER 25-27, 2013

Please join us **October 25-27, 2013 at the Hyatt in Dulles, Virginia for the first biennial vBSDCon event.** This exciting weekend will bring together members of the BSD community for a series of roundtable discussions, educational sessions, best practice conversations, and exclusive networking opportunities. See below for details on this industry weekend not to be missed:

AGENDA

- **Friday, October 25:** Evening Reception
- **Saturday, October 26:** General Session, Birds of a Feather Sessions
- **Sunday, October 27:** General Session, Breakout Sessions

WHO SHOULD ATTEND

- Developers
- Engineers
- Administrators
- Innovators

TOPICS

- PkgNG w/ Baptiste Daroussin
- A comprehensive look at bsdinstall with Devin Teske
- Netflix Demo/Presentation with Scott Long
- netmap with Luigi Rizzo
- Migration from GCC to LLVM/Clang with David Chisnall

REGISTRATION INFORMATION WILL BE SENT TO YOU IN MAY!

Questions? Please contact: eventsteam@verisign.com



VerisignInc.com

Log Space

Over time, logs will consume a lot of space in `/var/`. To reduce the disk usage, remove the logs. This can be done in a number of ways. If your company requires the logs be stored, then get a USB hard-drive and move the logs to the drive.

On a command-line, you could also run the `logrotate` command. This utility will compress old or large logs and give the system new, empty files to start writing more logs. To empty a single log, type a command like this:

```
echo "" > /var/logs/SOME_LOG
```

This will empty the log and keep the file without making a copy or compressing the file.

Before removing logs, check for any recent activity that should be noted.

Security

Viruses and hackers may try to destroy the system from the outside, or people physically near the server may cause harm. The server must be protected physically and at the software and network level. The server and network system can be secured at various levels in numerous ways.

For physical security, it is best to keep the server room locked and (if funding is sufficient) set up security cameras. Large companies with very important servers may want to consider hiring security guards. There are many other ways to secure the server physically, but that is beyond the scope of this magazine.

If local computers communicate with the server(s) via Wifi, the Wifi should use an encrypted signal and (if supported by the wifi router) enable MAC address filtering.

If a script needs to be added to the `/etc/rc*` system, thoroughly review the script and only place executables here from trusted developers. The `rc` utility starts scripts at boot time, at shutdown, and during other important events. If a virus gets installed here, it may be difficult to remove it and it can cause a lot of damage.

To see who has logged in, use the `last` command. This command may produce a long list of entries, so it may be better to pipe the output into `less`.

```
last | less
```

This command above will also show when the system is powered off. This can be helpful when figuring out when the system was last powered off. If the system loses power from the power supply, that will not be seen in this log.

The number one part of security that should not be neglected is anti-virus software. A popular open-source scanner is ClamAV. Beware though, anti-virus software can use up a lot of memory. While scanning, they can consume a portion of the CPU resources. Be sure to allow the virus scanner to scan the system and get definition updates after the company's closing time or during maintenance time. A script can be made to update the definitions and scan the system. Before leaving for the day, execute a script with contents like in Listing 1 below (remember to use root privileges). Techs may want to read the `man` pages for ClamAV and add the parameters that will best suit the system's needs (Listing 1).

Listing 1. Virus Scanner Script

```
#!/bin/sh
#Virus Scanner Script

sudo freshclam #Update definitions
sudo clamav 2>1 /root/scan_results.txt
shutdown -p now
```

Backups

Sometimes, the server may crash or lose data no matter how well it is maintained. To prevent permanent loss of data, the storage units should be backed up. Administrators will need extra storage devices. If the server has ten terabytes of data, then the backup storage should be the same amount or more. With the stability of USB devices and FreeBSD's excellent support for such hardware, this makes backups easier.

Plug in a USB external hard-drive. Use your preferred backup utility. Once finished, unmount and unplug the backup USB device and store it in a secure, dry, safe place. It may be best to store the drive in a fire-proof safe. Then, if the building or storage room catches on fire, the company will still have the data from the last backup.

Clonezilla is a live Linux disc that can be used like Ghost to make an image of the hard-drive to an external hard-drive.

For some, it may be best to only back up `/etc/`, `/home/`, `/root/`, and any other folders that may store important files that are needed that cannot be recovered through a fresh install. Keep a list or a storage device with all of the software installed after the original/last fresh install. Also, keep a copy of the installation disc of the preferred BSD operating system. Then, if the system must be reinstalled, the tech can install the BSD distro, install the applications, and then put the data and files back. Remember, when doing a fresh install, to reformat the hard-drive(s).

The filesystem may not have been formatted in a long time and the system crash may have been caused by, or caused, corruption of the filesystem.

If the system is completely ruined, use the “rescue mode” that is on most BSD installation discs. This recovery utility may help save the system and data. When the server boots from the disc, read the menu and press the button needed to initiate rescue mode.

Always make a second separate backup of the company’s databases and data. If the company’s data gets ruined from a hacker or for whatever reason, this backup will be helpful. Saving the company’s data may be more important than the server itself, so keep this data safe and make back-ups often. If the system is fast enough, make a script that will copy the data to an external or remote storage device during a lunch break or some other large break.

Important Rules

Here are some very important rules to follow when managing a server.

KISS

Keep It Simple Stupid. Do not write overly complicated scripts or use long strings of commands when a shorter or more simple script/command can be used. Making scripts/commands more complicated than they really should be can increase the chances that a coding mistake will be overlooked and do something to ruin the BSD system, especially if the script is executed in Root’s account. As another example, if a system needs to be updated only for the purpose of getting a new SCSI driver, then download and install that single driver (if possible) instead of updating the whole system. This rule also applies to deciding whether to install an application. If it is not needed, do not install it. Some installed programs can slow down the boot-up time.

NOTE

The above tip does not mean performing a task the lazy way. Perform tasks completely and correctly – just do not overdo it.

Do not use the name Root in vain – If a task can be completed on the server (or a workstation) using a regular account, do so. Only use Root when absolutely necessary, or else one accident can cause devastation to the server and network.

Know your BSD distros – Overall, BSD distributions are the same as far as file and application locations. However, some may be different or may not support some shell commands. Before running certain commands (like `rm`

and `mv`), make sure the files and directories you plan to manipulate exist in the location you are normally familiar with. Performing an action in the wrong directory can cause confusion when the system does not perform the actions it is assumed to complete.

If it is not broken, do not fix it – If an upgrade is not needed, then do not do it. Updating, editing, and replacing hardware can cause issues. It would be a waste of resources to do so on a system that does not need it when the system later has issues from an unneeded “fix”. For illustration, assume Bill upgrades the BSD operating system without a specific need. Before, everything ran well but now, there is some system conflict with the new system. The conflict could have been avoided if Bill had not upgraded the system. When a system needs a repair or update, then the potential risks and problems that may result from upgrading and repairing will be more worthwhile.

Certification Prep

For those of you wanting to get your BSD and/or LPI certifications, you must understand all of the server daemons and configuration files. You must thoroughly understand the great importance of system backups and how they are performed. Learning the location of important system and server logs will also help those wanting to earn such certifications. Obtaining these certifications will help admins get better jobs. Studying for such certifications will also give admins the knowledge they must have to sufficiently manage servers.

DEVYN COLLIER JOHNSON

Devyn Collier Johnson DevynCJohnson@Gmail.com has written many articles for Linux.org, wrote one article for the Full Circle Magazine on Clementine, and was the technical editor for McGraw Hill’s book Epub: From the Ground Up. The author has some experience and certifications in Linux/Unix systems. More about the author can be found here: <https://launchpad.net/~devyncjohnson-d>

Feel free to contact the author via email, Launchpad, Linux.org, or Google+ for questions or article requests.

Re-purposing an Abandoned Mac Mini

as a Wireless Router with OpenBSD

Don't let thousands of dollars worth of hardware that has been abandoned by Apple rot with old, unsupported software. Bring it into the present with OpenBSD.

What you will learn...

- How to work with OpenFirmware to install BSD
- Configuring a Mac Mini to work as a wireless router

What you should know...

- Basic BSD concepts
- Basic networking concepts

In June 2005, Apple announced that it would be switching their systems from PowerPC processors to Intel x86-based processors. Apple started offering these Intel models in January of 2006, and it was only a matter of time until the millions of PowerPC Macs in use were obsolete.

In August of 2009, Apple released Snow Leopard, OS X 10.6, their first Intel-only operating system. Their last PowerPC OS, 10.5 Leopard, was released in October 2007, and the last security fix for Leopard was released in May 2012. Adding insult to injury, that fix was Intel-only [1].

Many Mac users are therefore left with perfectly capable systems that in some cases cost them thousands of dollars, and they are unsatisfied running an outdated, insecure OS on them. PowerPC is currently a supported platform on OpenBSD [2] (it is also supported on NetBSD [3] and FreeBSD [4], but as a Tier 2 platform – which means that they are not fully supported), so I have chosen it to give new life to my PowerPC Mac.

Preparation

It is essential that the driver for the wireless adapter you intend to use for your router can operate in hostap mode, which means it can act as an access point (or base station) for other cards. As almost all models of Apple Airport wireless cards do not support HostAP mode, I am using an USB adapter for the LAN facing the wireless interface. There are not many USB adapters currently available that support hostap mode in OpenBSD, so it is very important to pick one with a chipset that uses the

`athn(4)`, `ural(4)` or `rum(4)` drivers [5]. For the example, I am using a TP-LINK TL-WN722N, which I picked up locally for \$15 USD.

Booting from CD

New World Macs use a boot system called OpenFirmware. OpenFirmware was originally developed by Sun Microsystems and is used in a variety of PowerPC systems from various manufacturers. It serves essentially the same purpose as the BIOS found in older PCs or the EFI found in newer Macs and PCs.

In order to be able to install OpenBSD on our Mac, we will need to access the OpenFirmware prompt. When booting, hold down `option-command-o-f` (or, on a non-Mac keyboard, `meta-alt-o-f`) and you will be brought to the OpenFirmware command line (Listing 1).

At the prompt, with the OpenBSD disk in the CD drive, assuming you are using version 5.3, type:

```
boot cd:,ofwboot 5.3/macppc/bsd.rd
```

This command tells the system to boot from the CD drive a file called `ofwboot`, the OpenBSD bootstrap program, in the root directory of the CD. Before the comma would typically come the partition number you'd like to boot to, but it is not applicable in the case of a CD. You are then passing to the `ofwboot` program the location of the BSD compressed RAMDISK kernel which contains an embedded filesystem with the installation tools.



Installing OpenBSD

Once the system has been booted, install OpenBSD as desired. If you have no intention of dual-booting with Mac OS X, choose an MBR partition map as then the bootstrap program will be installed automatically during installation, whereas for HFS disks ofwboot must be installed manually.

You can choose to configure the internal Ethernet device (`gem0`) during the installation or afterward.

Setting "Autoboot"

After installation, the system must be configured to automatically boot to our new OpenBSD installation. Boot the system to the OpenFirmware prompt once again and enter the following:

```
setenv auto-boot? True
setenv boot-device hd:,ofwboot
reset-all
```

We are telling OpenFirmware to automatically boot our system using the ofwboot bootstrap file on our hard drive, and telling it to save these changes into NVRAM with the reset-all command.

Enabling the Wireless Adapter

Once the newly installed system has booted, we will configure our wireless adapter. Assuming that the firmware has loaded correctly, you should see your adapter in the list when executing `ifconfig`. The adapter I'm using for this example uses the Atheros chipset and thus shows up as `athn0`. To set up a simple access point with basic WPA security, we would enter the following:

```
# ifconfig athn0 inet 172.16.0.1 netmask 255.255.255.0
media autoselect \
mediaopt hostap nwid miniwifi wpakey abcd1234 chan 1 up
```

We have now assigned our wireless adapter the IP address 172.16.0.1, set the `hostap` mode option, allowing our adapter to act as an access point, set the network ID to `miniwifi`, configured WPA with an example passphrase of `abcd1234` (you will want to use one much more secure) and brought the adapter up on channel 1.

Configuring the Router

Next, we are going to configure the system to allow routing network traffic through it. Configure IPv4 traffic to be forwarded between our interfaces:

```
sysctl net.inet.ip.forwarding=1
```

[GEEKED AT BIRTH]

IM Geek f



You can talk the talk.
Can you walk the walk?

[IT'S IN YOUR DNA]

- LEARN:**
- Advancing Computer Science
 - Artificial Life Programming
 - Digital Media
 - Digital Video
 - Enterprise Software Development
 - Game Art and Animation
 - Game Design
 - Game Programming
 - Human-Computer Interaction
 - Network Engineering
 - Network Security
 - Open Source Technologies
 - Robotics and Embedded Systems
 - Serious Game and Simulation
 - Strategic Technology Development
 - Technology Forensics
 - Technology Product Design
 - Technology Studies
 - Virtual Modeling and Design
 - Web and Social Media Technologies

Listing 1. OpenFirmware command line

```
Welcome to OpenFirmware

To continue booting, type "mac-boot" and press return.
To shut down, type "shut-down" and press return.

Release keys to continue!

Ok
0 >
```

Listing 2. dhcpd.conf

```
option domain-name "mini.mac"
option domain-name-servers 208.67.222.222,
                           207.67.220.220;

subnet 172.16.0.0 netmask 255.255.255.0 {
    option routers 172.16.0.1;

    range 172.16.0.100 172.16.0.150;
}
```

Listing 3. pf.conf

```
##Macros to simplify when we add more complex rules
    later

wired = "gem0"
wireless = "athn0"
icmp_types = "{ echoreq, unreachable}"

#Configure statistic logging for our wireless interface
set loginterface $wireless
#Allow lo0 to talk unrestricted
set skip on lo0

#Scrub incoming packets
match in all scrub (no-df)

#NAT for wireless clients
match out on egress from !(egress:network) to any
    nat-to (egress:0)

#Block all by default
block log all
#Let some traffic in
pass in inet proto { tcp udp } all
pass in inet proto icmp all icmp-type $icmp_types
```

References

- Apple Security Updates – <http://support.apple.com/kb/HT1222>
- OpenBSD Supported Platforms – <http://openbsd.org/plat.html>
- NetBSD Tier 2 Supported Platforms – <http://www.netbsd.org/ports/#ports-tier2>
- FreeBSD PowerPC Support – <http://www.freebsd.org/platforms/ppc.html>
- OpenBSD Wireless options and support – <http://openbsd.org/faq/faq6.html#Wireless>

This can be made persistent by uncommenting the appropriate line from `/etc/sysctl.conf`.

Next we will configure `dhcpd`. I set up a very simple `dhcpd.conf` to start: Listing 2.

With this configuration file, clients are served the internal domain name, `mini.mac`, and available DNS servers. In this case we're using `OpenDNS`, the network's subnet and its netmask, the default gateway, set to the address we assigned to our wireless adapter. I then set the range of addresses to be assigned to 172.16.0.100-150.

Next, to insure that `dhcpd` will start at boot, add the following line to your `/etc/rc.conf.local`:

```
dhcpd_flags="";
```

Restart `dhcpd` to make our changes take effect:

```
# /etc/rc.d/dhcpd restart
```

Configuring PF

The last thing we need to do for our simple router is to configure PF. We will create a very basic configuration NATing our internal addresses to the internet, allowing outbound traffic and some inbound traffic (Listing 3).

Lastly, we need to reload the modified PF configuration:

```
# pfctl -f /etc/pf.conf
```

Conclusion

You now have a perfectly good wireless router instead of an expensive doorstop. At this point, the router is very basic – the best next step would be to expand your PF rules to make it as useful and secure as possible. Also, you could continue configuring your own DNS server, installing Squid, or just about anything else that would be useful for your particular environment.

PATRICK ALLEN

Patrick Allen has been passionate about computers since he first sat down in front of his Commodore 64 many, many years ago. He lives in Colorado, USA with his wife and children.

Great Specials

On FreeBSD® & PC-BSD® Merchandise

Give us a call & ask about our
SOFTWARE BUNDLES

1.925.240.6652

\$39.95

FreeBSD 9.1 Jewel Case CD Set
or FreeBSD 9.1 DVD

\$29.95

PC-BSD 9.1 DVD

\$49.95

The PC-BSD 9.0 Users Handbook
PC-BSD 9.1 DVD



\$99.95

The FreeBSD CD or DVD Bundle

Inside each CD/DVD Bundle, you'll find:
FreeBSD Handbook, 3rd Edition
Users Guide FreeBSD Handbook, 3rd Edition, Admin Guide
FreeBSD 9.1 CD or DVD set
FreeBSD Toolkit DVD

Stylish Dress Attire
Look Your Professional Best



Comfy Apparel
Stay Warm in Zip Ups & Pullovers

T-Shirts
Lots of Styles to Choose From

FreeBSD 9.1 Jewel Case CD/DVD.....\$39.95

CD Set Contains:

- Disc 1** Installation Boot LiveCD (i386)
- Disc 2** Essential Packages Xorg (i386)
- Disc 3** Essential Packages, GNOME2 (i386)
- Disc 4** Essential Packages (i386)

FreeBSD 9.0 CD.....\$39.95

FreeBSD 9.0 DVD.....\$39.95

FreeBSD Subscriptions

Save time and \$\$\$ by subscribing to regular updates of FreeBSD

FreeBSD Subscription, start with CD 9.1.....\$29.95

FreeBSD Subscription, start with DVD 9.1.....\$29.95

FreeBSD Subscription, start with CD 9.0.....\$29.95

FreeBSD Subscription, start with DVD 9.0.....\$29.95

PC-BSD 9.1 DVD (Isotope Edition)

PC-BSD 9.1 DVD.....\$29.95

PC-BSD Subscription.....\$19.95

The FreeBSD Handbook

The FreeBSD Handbook, Volume 1 (User Guide).....\$39.95

The FreeBSD Handbook, Volume 2 (Admin Guide).....\$39.95

The FreeBSD Handbook Specials

The FreeBSD Handbook, Volume 2 (Both Volumes).....\$59.95

The FreeBSD Handbook, Both Volumes & FreeBSD 9.1.....\$79.95

PC-BSD 9.0 Users Handbook.....\$24.95

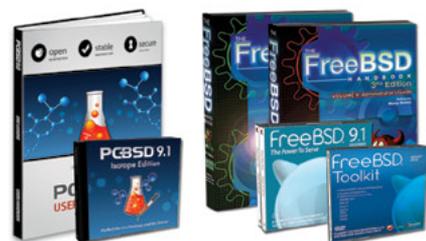
BSD Magazine.....\$11.99

The FreeBSD Toolkit DVD.....\$39.95

FreeBSD Mousepad.....\$10.00

FreeBSD & PCBSD Caps.....\$20.00

BSD Daemon Horns.....\$2.00



Bundle Specials!
Save \$\$\$

Just Plain Fun
Mousepads & Novelty Horns



BSD Magazine
Available Monthly



For even MORE items
visit our website today!

www.FreeBSDMall.com

Monit

Monitoring solution for enterprise and SOHO servers with FreeBSD

The state of the network services offered by a company is the business card with which it presents itself to the world. Thanks to constant monitoring, it is possible to understand how and where to improve the infrastructure of the network in real time and detect any abnormalities.

What you will learn...

- In this paper we will learn to setup and manage a Monitoring server based on monit.

What you should know...

- Basics about how network services work and basic BSD networking setup.

Monit is software that should not be missing in the kit of a network administrator as it automatically controls and manages the applications on the server side in order to ensure that they are consistently active, for example, by testing the size of the files and permissions. In particular, Monit provides system administrators with an excellent control tool also equipped with a web interface. This allows you to manage the state of the system and processes using either a standard web browser or from the command line.

In this article, we will carry out the monitoring of a web server like Apache or Nginx with MySQL, SSH, and cron.

Installing a FAMP Environment: the easy way

This section describes how to set up Apache, MySQL, PHP and phpMyAdmin on a server running FreeBSD. The article was written for the software versions below but is likely to work on newer versions without too much difficulty.

Installing MySQL:

```
cd /usr/ports/databases/mysql155-server
make BUILD_OPTIMIZED=yes BUILD_STATIC=yes
make install clean
```

Open `/etc/rc.conf` with your favourite text editor and add the line shown below. This will ensure mysql is enabled and starts on boot.

```
mysql_enable="YES"
```

Start mysql manually to avoid having to reboot now by typing:

```
/usr/local/etc/rc.d/mysql-server start
```

Set a password for the MySQL root user by executing the command, substituting your own password in place of `new-password`:

```
/usr/local/bin/mysqladmin -uroot
password `new-password`
```

And now MySQL is installed. Let's start to install Apache.

```
cd /usr/ports/www/apache22
make install clean
```

It's a good idea to disable the two DAV options, if you don't need them, when prompted.

Open `/etc/rc.conf` with your favourite text editor and add this line shown below to start Apache on boot.

```
apache22_enable="YES"
```

Installing PHP

```
cd /usr/ports/lang/php5
make install clean
```

It's very important to verify that the APACHE (Build Apache module) option is ticked and leave all other options as default before selecting OK.

```
cd /usr/ports/lang/php5-extensions
make install clean
```

Install the php.ini file:

```
cp /usr/local/etc/php.ini-dist /usr/local/etc/php.ini
```

Edit the Apache configuration file `/usr/local/etc/apache22/httpd.conf` by adding the following lines:

```
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
```

Change the line: `DirectoryIndex index.html ...to`

```
DirectoryIndex index.php index.html ...
```

Enable language settings by searching for the line:

```
#Include etc/apache22/extra/httpd-languages.conf
```

and removing the # comment mark so it reads:

```
Include etc/apache22/extra/httpd-languages.conf
```

Edit the language settings file (`/usr/local/etc/apache22/extra/httpd-languages.conf`) and add the following line at the end of the file:

```
AddDefaultCharset On
```

Start Apache using the startup script:

```
/usr/local/etc/rc.d/apache22 start
```

And you're done! Apache with PHP and MySQL are installed.

After the FAMP environment, let's install Monit

Installing Monit is really very simple. On Free / Open BSD, just log on to the terminal with administrator rights and type:

```
# pkg_add -r -v monit
```

To verify that the installation is successful, at the terminal simply, run the command:

```
# monit status
```

which should return the statistics for general use of the system.

A simple configuration

It's really easy to create effective monitoring systems with the use of Monit. By default, it checks every two minutes if a service is active and stores the information in the log file `/var/log/monit.log`.

However, it is possible to change any of the settings by going to correctly fill in the fields in the main configuration file, respectively `/etc/monit/monitrc` in Debian based distributions and `/usr/local/etc/monitrc` on FreeBSD.

Monit natively provides a web server that responds on port 2812. To enable it properly, just edit the configuration file and restart the service with a "monit reload" from a terminal as root. Below is a simple example that allows access to the server from the local network 192.168.200.50 192.168.200.0/24 with username "admin" and password "monit".

```
set httpd port 2812
    use address 192.168.200.50
    allow 0.0.0.0/0.0.0.0
    allow admin:monit
    allow @monit
    allow @users readonly
```

```
root@bsdMonitoringSrv:/root # monit status all
The Monit daemon 5.5 uptime: 6m

Process 'mysql'
status          Running
monitoring status Monitored
pid             1428
parent pid     1437
uptime         17m
children       9
memory kilobytes 15892
memory kilobytes total 15892
memory percent  5.8%
memory percent total 6.8%
cpu percent     0.1%
cpu percent total 0.1%
port response time
data collected  0.006s to 127.0.0.1:3306 [DEFAULT via TCP]
Sun, 25 Aug 2013 17:12:19

Process 'sshd'
status          Running
monitoring status Monitored
pid             797
parent pid     1
uptime         1h 0m
children       3
memory kilobytes 3908
memory kilobytes total 12884
memory percent  1.2%
memory percent total 5.2%
cpu percent     0.6%
cpu percent total 0.6%
port response time
data collected  0.611s to localhost:22 [SSH via TCP]
Sun, 25 Aug 2013 17:12:19

Process 'httpd'
status          Running
monitoring status Monitored
pid             798
parent pid     1
uptime         1h 0m
children       6
memory kilobytes 4968
memory kilobytes total 34872
memory percent  2.6%
```

Figure 1. Monit, services console monitoring

The last two lines indicate that the users of the group “monit” have full read/write access to the system, while users of the “users” group have read-only access.

Now, with the browser on the IP or domain name of the server and logging in with the credentials created before, you will see a screen like this (If one wants to enable SSL access, one must create an ad hoc certificate) (Figure 1).

The simplest configuration that can be set relates to the monitoring of the services running on the server itself. In fact, by properly editing the `/usr/local/etc/monitrc`, one can be sure to keep them always “up”. If the services active are Apache, MySQL and Nginx, they would be automatically tracked and restarted in case of block, and it would be possible to check the status from both the web interface and text-based interface (Listing 1).

To check the correct syntax of the configuration file simply run the command:

```
monit -t
```

Once the output of the command returns “Control file syntax OK”, the system is set up and ready to manage network services. To enable all tracking systems, simply run:

```
monit start all
```

In the heart of Monit configuration file

An entry for the control of a generic process in the file `/usr/local/etc/monitrc` has the following syntax:

```
check process PROCESSNAME
    with pidfile PIDFILENAME-WITHABSOLUTE-PATH
    start = STARTUP-SCRIPT
    stop = STOP-SCRIPT
```

This item, in particular, checks the status of the local server cron daemon:

```
check process crond
    with pidfile /var/run/crond.pid
    start = "/etc/init.d/cron start"
    stop = "/etc/init.d/cron stop"
```

However, it is always possible to refer to the monit man page to get the complete set of instructions and options to best configure the control system. There are two options you should activate in any case, and that is the logging system in the file `/usr/local/etc/monitrc` (In this case, everything is branded on the general system log file):

```
set logfile syslog facility log_daemon
```

and automatically start monit as a system daemon, which is accomplished by editing the file `/etc/rc.conf` and adding the item:

```
monit_enable="YES"
```

Enabling monit as a system service (with updates every minute):

```
set daemon 60
```

A further change to the configuration of the logging via syslog lets you use a dedicated file for easier management of alerts:

```
set logfile syslog facility log_daemon
set logfile /var/log/monit.log
```

Configure your email server to send email notification:

Listing 1. `/usr/local/etc/monitrc`

```
## Apache:
check process apache with pidfile /var/run/httpd.pid
    start program = "/usr/local/etc/rc.d/apache22
        start" with timeout 60 seconds
    stop program = "/usr/local/etc/rc.d/apache22 stop"

## MySQL
check process mysqld with pidfile /var/run/
    mysqld/"${hostname -f} ".pid
    start program = "/usr/local/etc/rc.d/mysql-server start"
    stop program = "/usr/local/etc/rc.d/mysql-server stop"

## Nginx
check process nginx with pidfile /var/run/nginx.pid
    start program = "/usr/local/etc/rc.d/nginx start"
    stop program = "/usr/local/etc/rc.d/nginx stop"
```

Listing 2. Example of Apache service management

```
check process httpd with pidfile /var/run/httpd.pid
group www
    start program = "/usr/local/etc/rc.d/apache22
        start" with timeout 60 seconds
    stop program = "/usr/local/etc/rc.d/apache22 stop"
if failed host 127.0.0.1 port 80 protocol http then restart
if 5 restarts within 5 cycles then timeout
```

```
set mailserver mail.yourmailserver.tld
Set email format such as from email
set mail-format { from: monit@yourmailserver.tld
subject: $$SERVICE $EVENT at $DATE
message: Monit $ACTION $SE
VICE at $DATE on $HOST: $DESCRIPTION.
}
```

Example of Apache service management: Listing 2.

Entries in a control call are:

- **check process httpd with pidfile /var/run/httpd.pid:** You specify the pid file httpd.pid and its “daemon name”
- **group www:** We specify the www group, with permissions for startup / shutdown of the service.
- **start program = “/usr/local/etc/rc.d/apache22 start”:** Startup script.
- **stop program = “/usr/local/etc/rc.d/apache22 stop”:** Shutdown script.
- **if failed host 127.0.0.1 port 80:** Server IP address and listening port (80).
- **protocol http then restart:** The web server restarts if it is not possible to reach the IP and port specified.
- **if 5 restarts within 5 cycles then timeout:** Five attempts to restart have been made. If monit can not run, it takes the timeout condition.

Below is an example of service management MySQL and ssh on the local server: Listing 3.

To start and test monit, the easiest method is to “kill a process”; the daemon will restart and the system log will store the event. Monit will trace the status of cron as “inactive” and will start the next operation without requiring intervention by a system administrator.

Taking a look at the log file, we can verify that cron was revived by monit after being “killed” as shown in Figure 2.

The whole server room in a click

The web server integrated in Monit provides different levels of access: simple (no control over the LAN IP and no request for username / password), and standard (requested username / password and a block of IP addresses from which you can access). You can optionally integrate an SSL certificate (Figure 4).

Once logged in, we see a summary screen with the status of the servers and services running on it. Clicking on the individual links will access details of all the data of interest (Figure 5).

From this screen, you can see the detail of the data to the server, enable and disable the monitoring real time,

discover in addition to the system load average, even the CPU, RAM and SWAP, and get to know when the data was collected, perhaps for statistical purposes (Figure 6).

This screen allows you to check, in one fell swoop, all the details of each service user group by pid to the process owner. You can also start / stop / restart the service itself and enable / disable the monitoring in addition to the usual data, including pid and CPU and RAM usage.

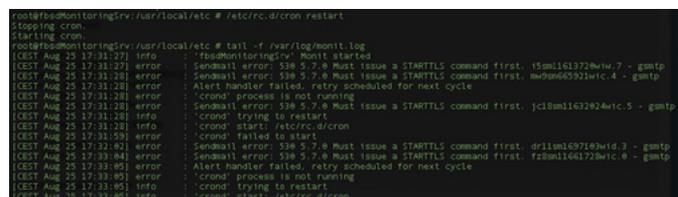


Figure 2. Monit, service cron restart

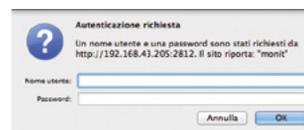


Figure 3. Monit, ACCESS TO THE SYSTEM BY WEB

System	Status	Load	CPU	Memory	Swap
bsdMonitoringSvc	Running	[0.00] [0.00] [0.00]	0.0%us, 0.0%ky	15.6% [30884 kB]	0.0% [0 kB]

Process	Status	Uptime	CPU Total	Memory Total
mysql	Running	47m	0.0%	6.9% [17100 kB]
sshd	Running	1h 30m	0.0%	7.6% [18600 kB]
httpd	Running	1h 30m	0.0%	14.2% [34800 kB]
cron	Running	2m	0.0%	0.5% [1400 kB]

Figure 4. Monit, A GENERAL OVERVIEW

Listing 3. Example of service management MySQL and ssh on the local server

```
check process mysqld with pidfile /var/run/
                                mysqld/"${hostname -f}".pid

group mysql
    start program = "/usr/local/etc/rc.d/mysql-server start"
    stop program = "/usr/local/etc/rc.d/mysql-server stop"
    if failed host 127.0.0.1 port 3306 then restart
    if 5 restarts within 5 cycles then timeout

check process sshd with pidfile /var/run/sshd.pid
start program "/etc/rc.d/sshd start"
stop program "/etc/rc.d/sshd stop"
if failed host 127.0.0.1 port 22 protocol ssh then restart
if 5 restarts within 5 cycles then timeout
```

Administer Vsftpd with Monit – an example of missing pidfile

Some services, such as the well-known server Vsftpd, do not lean against the “pidfile” to interact with the system, and it is then complex to manage them with monit. Moreover, services are so widespread that a solution, even if partial, can be very convenient. The following link may provide cues suitable to solve the problem which is beyond the scope of this article (<http://serverfault.com/questions/270316/monit-check-process-without-pidfile>).

```
check process vsftpd
    matching vsftpd
    start program = "/etc/init.d/vsftpd start"
    stop program = "/etc/init.d/vsftpd stop"
```

Finally we must add the library of pre-configured vsftpd to the monit control file and then restart the service for the changes to take effect.

Advanced Configurations from the heart of the server rooms

Below here is a small library of configurations to be added to your monitoring system. In particular, they refer to the location of the pidfile and the methods to start / stop the services (Listing 4).

Of course, there are many other services that can be monitored with this tool, but this base should be sufficient to take the first steps. From this point on, you're only limited by your imagination (Figure 7).

ANTONIO FRANCESCO

Antonio Francesco Gentile lives in Italy, Calabria and is a software and network engineer. He works for the National Research Institute ICAR as network manager, with the Culture Lab (<http://culture.deis.unical.it>) in the Department of Telematics at the University of Calabria, the computer science associations, Hacklab Cosenza (<http://hacklab.cosenzainrete.it/>) and Verde Binario (<http://www.verdebinario.org/>), and is a freelance columnist for Italian magazines, Linux&C (<http://www.oltrelinux.com/>) and Linux Magazine (<http://www.linuxmagazine.it/>).

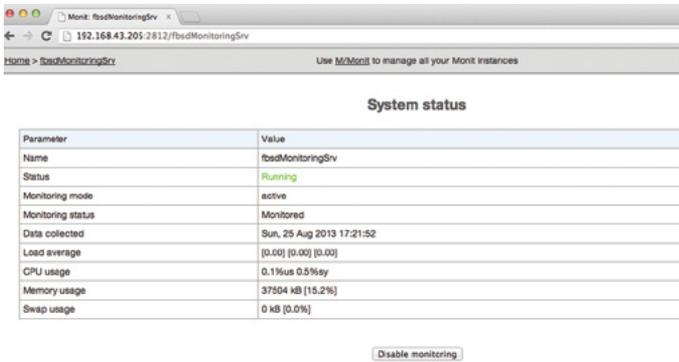


Figure 5. Monit, BEFORE THE SERVER

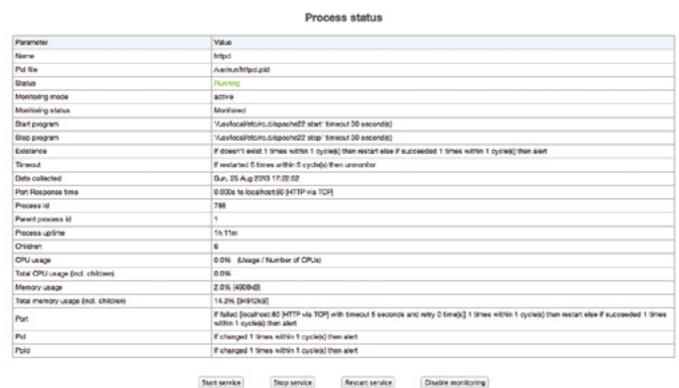


Figure 6. Monit, THEN SERVICES

```
root@fbsdMonitoringSrv: /usr/local/etc # cat monitrc
set daemon 60
set httpd port 2812 and
    use address localhost
    use address 192.168.43.205
    allow localhost
    allow 192.168.0.0/16
    allow admin:monit
    allow @monit
    allow @users readonly

set logfile syslog facility log_daemon
set logfile /var/log/monit.log

set mailserver smtp.gmail.com,
    mail.yourcompany.tld,
    localhost
set eventqueue
    basedir /var/monit
    slots 100

set mail-format {
    from: monit@$HOST
    subject: monit alert -- $EVENT $SERVICE
    message: $EVENT Service $SERVICE
        Date: $DATE
        Action: $ACTION
        Host: $HOST
        Description: $DESCRIPTION

    Your faithful employee,
    Monit
}

set alert antofrage@gmail.com

# Monitoring Mysql Service
check process mysql with pidfile /var/db/mysql/fbsdMonitoringSrv.pid
group wheel
start program = "/usr/local/etc/rc.d/mysql-server start"
stop program = "/usr/local/etc/rc.d/mysql-server stop"
if failed host 127.0.0.1 port 3306 then restart
if 5 restarts within 5 cycles then timeout
```

Figure 7. Monit, extract of a production server monitrc file

Listing 4. Advanced Configurations from the heart of the server rooms

```

## Syslogd (system logfile daemon)
check process syslogd with pidfile /var/run/syslogd.pid
  start program = "/etc/rc.d/syslogd start"
  stop program = "/etc/rc.d/syslogd stop"
  if 5 restarts within 5 cycles then timeout
check file syslogd_file with path /var/log/syslog
  if timestamp > 65 minutes then alert

## Net-SNMP (SNMP agent)
check process snmpd with pidfile /var/run/snmpd.pid
  start program = "/usr/local/etc/rc.d/snmpd.sh start"
  stop program = "/usr/local/etc/rc.d/snmpd.sh stop"
  if failed host 192.168.1.1 port 161 type udp then
    restart
  if failed host 192.168.1.1 port 199 type tcp then
    restart
  if 5 restarts within 5 cycles then timeout

## NTP (time server)
check process ntpd with pidfile /var/run/ntpd.pid
  start program = "/etc/rc.d/ntpd start"
  stop program = "/etc/rc.d/ntpd stop"
  if failed host 127.0.0.1 port 123 type udp then alert
  if 5 restarts within 5 cycles then timeout

## Bind (chrooted)
check process named with pidfile /var/named/chroot/var/
  run/named/named.pid
  start program = "/etc/rc.d/named start"
  stop program = "/etc/rc.d/named stop"
  if failed host 127.0.0.1 port 53 type tcp protocol
    dns then alert
  if failed host 127.0.0.1 port 53 type udp protocol
    dns then alert
  if 5 restarts within 5 cycles then timeout

## Squid (http/ftp proxy)
check process squid with pidfile /usr/local/squid/logs/
  squid.pid
  group www
  start program = "/usr/local/etc/rc.d/squid start"
  stop program = "/usr/local/etc/rc.d/squid stop"
  if failed host 192.168.1.1 port 3128 then restart
  if 5 restarts within 5 cycles then timeout
  depends on squid_bin
  depends on squid_rc

check file squid_bin with path /usr/local/bin/squid
  group www
  if failed checksum then unmonitor
  if failed permission 755 then unmonitor
  if failed uid root then unmonitor
  if failed gid root then unmonitor
check file squid_rc with path /usr/local/etc/rc.d/squid
  group www
  if failed checksum then unmonitor
  if failed permission 755 then unmonitor
  if failed uid root then unmonitor
  if failed gid root then unmonitor

## Postfix (mail server)
check process postfix with pidfile /var/spool/postfix/pid/
  master.pid
  group mail
  start program = "/usr/local/etc/rc.d/postfix start"
  stop program = "/usr/local/etc/rc.d/postfix stop"
  if failed port 25 protocol smtp then restart
  if 5 restarts within 5 cycles then timeout
  depends on postfix_rc
check file postfix_rc with path /usr/local/etc/rc.d/
  postfix
  group mail
  if failed checksum then unmonitor
  if failed permission 755 then unmonitor
  if failed uid root then unmonitor
  if failed gid root then unmonitor

## Dovecot (imap secure server)
check process dovecot with pidfile /var/run/dovecot/
  master.pid
  start program = "/usr/local/etc/rc.d/dovecot start"
  stop program = "/usr/local/etc/rc.d/dovecot stop"
  group mail
  if failed host mail.yourdomain.tld port 993 type
    tcpssl sslauto protocol imap for 5
    cycles then restart
  if 3 restarts within 5 cycles then timeout
  depends dovecot_init
  depends dovecot_bin
check file dovecot_init with path /usr/local/etc/rc.d/
  dovecot
  group mail
check file dovecot_bin with path /usr/sbin/dovecot
  group mail

```

FreeBSD Programming Primer – Part 8

In the eighth part of our series on programming, we will refine our JQuery menu and start building a user friendly interface to add content.

What you will learn...

- How to configure a development environment and write HTML, CSS, PHP and SQL code

What you should know...

- BSD and general PC administration skills

In the previous article, we implemented a menu using the JQuery library. Looking at menu.inc, we see the menu is “hard coded” with a top level menu Home, and sub menu’s Pages, News and FAQ’s. To make our CMS user friendly, ideally we would store the menu values in a database table that we could access and amend from a web form (Listing 1 and Figure 1).

Rather than building a custom page for each table, it would be good practice to design a set of global functions (e.g. sign on, retrieve fields, save fields etc.) and design a

template that we could change on a per table / form basis. We could then quickly build forms to modify each type of content. We also need to tweak the CSS for our dropdown menu. At the moment with the default CSS, the menu is floating to the left hand side. We will modify this to accommodate a wider menu with more options.

Step 1

For the initial testing, we will hand code a menu in menu.inc and add a few placeholders. Once we are happy with the CSS, we will then add this to a database table and add our forms. In the next article, we will write the code to extract the menu values and fire them into JQuery.



Figure 1. Original JQuery menu

Listing 1. menu.inc

```
<ul id="menu">
<li>
<a href="/">Home</a>
<ul>
<li><a href="/page/1">Pages</a></li>
<li><a href="/news/1">News</a></li>
<li><a href="/faq/1">FAQ' s</a></li>
</ul>
</li>
</ul>
```



Figure 2. JQuery menu horizontal

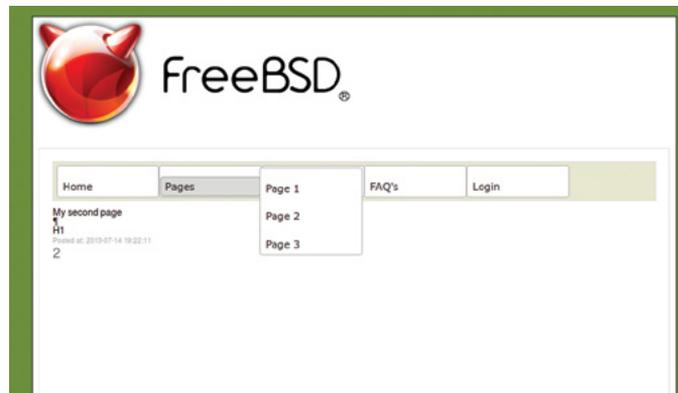


Figure 3. JQuery menu with drop down menu

Listing 2. Replacement JQuery menu

```
<div id="jquerymenu">
  <ul id="top-menu-home">
    <li><a href="/">Home</a></li>
  </ul>

  <ul id="top-menu-pages">
    <li><a href="">Pages</a>
      <ul>
        <li><a href="/page/1">Page 1</a></li>
        <li><a href="/page/2">Page 2</a></li>
        <li><a href="/page/3">Page 3</a></li>
      </ul>
    </li>
  </ul>

  <ul id="top-menu-news">
    <li><a href="">News</a>
      <ul>
        <li><a href="/news/1">News 1</a></li>
        <li><a href="/news/2">News 2</a></li>
        <li><a href="/news/3">News 3</a></li>
      </ul>
    </li>
  </ul>

  <ul id="top-menu-faq">
    <li><a href="">FAQ's</a>
      <ul>
        <li><a href="/faq/1">FAQ 1</a></li>
        <li><a href="/faq/2">FAQ 2</a></li>
        <li><a href="/faq/3">FAQ 3</a></li>
      </ul>
    </li>
  </ul>
</div>
```

```
<ul id="top-menu-user">
  <li><a href="/login.php">Login</a></li>
</ul>
</div>
```

Listing 3. preload.js

```
function globalmenu() {
    $(function() {$( "#top-menu-home" ).menu();});
    $(function() {$( "#top-menu-pages" ).menu();});
    $(function() {$( "#top-menu-news" ).menu();});
    $(function() {$( "#top-menu-faq" ).menu();});
    $(function() {$( "#top-menu-user"
        ).menu();});
}
}
```

Listing 4. global.css

```
#jquerymenu {
    border: 1px solid #DADADA;
    margin-bottom: 10px;
    height: 48px;
    padding: 5px;
    background-color: #e8e7cf;
}

.ui-menu{
    float: left;
}
```

Listing 5. *create the menus table*

```
CREATE TABLE `menus` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `group` varchar(12) NOT NULL,
  `menutitle` varchar(12) NOT NULL,
  `titleurl` varchar(12) DEFAULT NULL,
  `submenutitle` varchar(50) DEFAULT NULL,
  `submenutitleurl` varchar(50) DEFAULT NULL,
  `order` int(2) NOT NULL DEFAULT '0',
  `enabled` int(1) NOT NULL DEFAULT '1',
  `timestamp` timestamp NOT NULL DEFAULT CURRENT_
    TIMESTAMP,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Listing 6. *populate the menus table*

```
INSERT INTO `menus` (`id`,`group`,`menutitle`,`titleurl`,
  `submenutitle`,
  `submenutitleurl`,`order`,`enabled`,`timestamp`) VALUES
(1,'jquerymenu','Home','/',NULL,NULL,1,1,'2013-09-02
  17:50:05'),
(2,'jquerymenu','Pages',NULL,NULL,NULL,2,1,'2013-09-02
  17:54:58'),
(3,'jquerymenu','Pages',NULL,'Page 1','/
  page/1',1,1,'2013-09-02 17:56:33'),
(4,'jquerymenu','Pages',NULL,'Page 2','/
  page/2',2,1,'2013-09-02 17:57:28'),
(5,'jquerymenu','Pages',NULL,'Page 3','/
  page/3',3,0,'2013-09-02 17:58:08');
```

Listing 7. *amendcontentpage.php*

```
<?php
```

```
require_once 'includes/cms.inc';
require INCLUDES . 'content.inc';
require INCLUDES . 'core.inc';
require INCLUDES . 'html.inc';
require INCLUDES . 'mysql.inc';

// SQL statements

$sql[0] = "SELECT COUNT(DISTINCT TABLE_NAME) FROM
  INFORMATION_SCHEMA.COLUMNS
  WHERE table_schema = 'freebsdcms'
  AND TABLE_NAME = '---P0---'";
$sql[1] = "SELECT
  TABLE_NAME,COLUMN_NAME,COLUMN_DEFAULT,IS_
  NULLABLE,DATA_TYPE,
  CHARACTER_MAXIMUM_LENGTH
```

```
FROM INFORMATION_SCHEMA.COLUMNS
WHERE table_schema = 'freebsdcms'
AND TABLE_NAME = '---P0---'
ORDER BY table_name, ordinal_position";
```

```
// The tables we will allow the user to edit via this
  form
```

```
$tables[] = "faqs";
$tables[] = "menus";
$tables[] = "news";
$tables[] = "pages";
```

```
// Fields that are automatically assigned via a default
  value in MySQL table
```

```
// definition
```

```
$skiplist[] = "id";
$skiplist[] = "timestamp";
```

```
////////////////////////////////////
////////////////////////////////////
////////////////////////////////////
```

```
// Build the page up to the body tag
```

```
outfile(TEMPLATES . 'header.inc');
echo wraptag('title', 'Content Input');
echo HEAD;
echo BODY;
```

```
// Page control logic
```

```
if(isset($_POST["table"])){
```

```
// User has not selected a table or we are testing
  their result
```

```
$t = $_POST["table"];
```

```
if (!in_array($t, $tables)) {
```

```
// If the table is not on allowed list, bail to
  the first page
```

```
build_page_1($tables);
```

```
}else{
```

```

// Check selected table is valid

$s = $sql[0];

// Replace the marker in the SQL statement with
// the chosen value
$s = str_replace ( '---P0---' , $t , $s );

$result = mysql_select($s);
$valid_table_count = $result['COUNT(DISTINCT
    TABLE_NAME)'];

if($valid_table_count == 1){

    // Valid table selected - present form to
    // edit data
    build_page_2($t,$sql,$skiplist);

}else{

    // Send user to first page
    build_page_1($tables);

}

}

}elseif(isset($_POST["update"])){

    // Save the input. As we have not validated this,
    // just display for now

    build_page_3($_POST);

}else{

    // Invalid value - return to start

    build_page_1($tables);

}

////////////////////////////////////
////////////////////////////////////
////////////////////////////////////

function build_page_1($tables){

    // HTML form definition

    echo '<div id="content">';
    echo '<div id="php">';
    echo '<div id="h1">1: Select content</div>';
    echo '<form action="amendcontent.php" method="post">';
    echo '<select name="table">';

    foreach ($tables as $t){

        // $tables is an array - split each value out

        echo '<option value="'. $t. '">'. $t. '</option>';

    }

    // Finish form and add footer

    echo '</select>';
    echo '<input type="submit" value="Select content to
        edit">';

    echo '</form>';
    echo '</div></div>';
    echo '<div id="licence">';
    echo '<a href="licence.txt" title="Copyright and licence
        details">Copyright &copy; 2013 Rob
        Somerville me@merville.co.uk</a>';

    echo '</div>';

}

function build_page_2($t,$sql,$skiplist){

    // HTML form

    echo '<div id="content">';
    echo '<div id="php">';
    echo '<div id="h1">2: Edit <?php echo $t;
        ?>&nbsp;content</div>';
    echo '<form action="amendcontent.php" method="post">';

    // Get the schema for that particular table

    $s = $sql[1];
    $s = str_replace ( '---P0---' , $t , $s );

    $result = mysql_fetchrows($s);
    $divstart = '<div class="inputname">';

```

```

echo '<input type="hidden" name="update"
      value="' . $t. "'>';

foreach($result as $row) {

    // Loop through each field and build the form
    // fields depending on the field //
    // type

    $field = $row[1];
    $fieldtype = $row[4];

    if (!in_array($field, $skiplist)) {

        if($fieldtype == "varchar"){

            echo $divstart . ucfirst($field). '</div><input class="varchar"
            type="text" name="' . $field. '"><br />';

        }elseif($fieldtype == "int"){

            echo $divstart . ucfirst($field). '</div><input class="int"
            type="text" name="' . $field. '"><br />';

        }elseif($fieldtype == "text"){

            echo $divstart . ucfirst($field). '</div><textarea rows="10" cols="30"
            class="textarea" name="' . $field. '"></div><textarea><br />';

        }else{

            // Shouldn't get here

            echo 'Error field(' . $field. ') \'.
            $row[2].'|'. $row[3]. '|'.
            $row[4].'|'. $row[5]. '|'. <br
            />';

        }

    }

}

// Finish form and add footer

echo '</select>';
echo '<input type="submit" value="Save changes">';
echo '</form>';
echo '</div></div>';
echo '<div id="licence">';
echo '<a href="licence.txt" title="Copyright and licence
      details">Copyright &copy; 2013
      Rob Somerville me@merville.co.uk</a>';
echo '</div>';

}

function build_page_3($post) {

    // HTML

    echo '<div id="content">';
    echo '<div id="php">';
    echo '<div id="hl">3: Save content</div>';
    echo '<ul>';

    foreach($post as $key => $value) {

        // Just loop through and dump out values - we need
        // to validate before adding to DB

        echo '<li><b>'. $key. '</b>: \'. $value. '</li>';

    }

    // End of form

    echo '</ul><br />';
    echo '<a href="amendcontent.php">Return to add content</a>';

    echo '</div></div>';
    echo '<div id="licence">';
    echo '<a href="licence.txt" title="Copyright and licence
      details">Copyright &copy; 2013 Rob
      Somerville me@merville.co.uk</a>';
    echo '</div>';

}

```

Replace the code in (Listing 1) with the code in (Listing 2) and modify preload.js as well as global.css to match (Listing 3) and (Listing 4). This will provide the menu as shown in (Figure 2 & 3).

Add jquery support for each menu: Listing 3. Add some additional CSS so that the individual menus line up: Listing 4.

Step 2 – Create the menus table

In MySQL, create the menus table (Listing 5). Populate with some basic menus (Listing 6).

Step 3 – Build the amendcontent page

The amendcontent page is a PHP script that allows the user to add new content to the CMS. As we have not validat-

Listing 8. additions to global.css

```
#php {
    min-height: 640px;
    margin-top: 160px;
}

.varchar {
    background-color: #ced8f8;
    border: 1px solid #FFF;
}

.int {
    background-color: #cef8f5;
    border: 1px solid #FFF;
}

.textarea {
    background-color: #e3f3dc;
    border: 1px solid #FFF;
}

.inputname {
    color: tomato;
    font-size: 12px;
    width: 100px;
    float: left;
    font-weight: bold;
}
```



Figure 4. Select the table to edit



Figure 5. Add your data



Figure 6. What will be saved

Useful links

- JQuery UI source – <http://jqueryui.com/resources/download/jquery-ui-1.10.3.zip>
- JQuery menu reference – <http://jqueryui.com/menu>
- PHP manual – <http://php.net/manual>

ed the user input yet, we'll just capture the input for now. Create a new PHP file called amendcontent.php in the root directory where index.php is already saved (Listing 7).

We need to add a small modification to global.css to line up the fields (Listing 8). Now visit <http://yoursite/amend-content.php> and you will have a dynamic form ready to save data to any table in the CMS. See (Figure 4-6).

In the next article

We will use the data from the menu tables to populate the JQuery menus and write some validation code for the user input prior to saving to the database.

ROB SOMERVILLE

Rob Somerville has been passionate about technology since his early teens. A keen advocate of open systems since the mid-eighties, he has worked in many corporate sectors including finance, automotive, airlines, government and media in a variety of roles from technical support, system administrator, developer, systems integrator and IT manager. He has moved on from CP/M and nixie tubes but keeps a soldering iron handy just in case.

A Closer Look at the Changes in PC-BSD/TrueOS 9.2. Part 2

Directory encryption using PEFS

PC-BSD

Last month we took a look at how PC-BSD is implementing ZFS boot-environments, which can be a life-saver for both servers and desktops. This month we will be looking at how PC-BSD uses the PEFS kernel level file system module to automatically encrypt your home directory and its contents, and how you can manually run PEFS for other sensitive data.

Starting in PC-BSD 9.2, the default encryption provider has been switched to PEFS, which has also been merged into the base operating system as the `/boot/kernel/pefs.ko` kernel module, the `/usr/sbin/pefs` command, and various libraries. The PEFS system is a kernel level stacked cryptographic file-system authored by Gleb Kurtsov. Because it is provided as a kernel module, it does not require any user-level daemons to function, and can run on top of existing file-systems such as ZFS in the case of PC-BSD or TrueOS. It includes other nice features such as random cipher tweak values on a per-file basis, support for AES / Camellia in XTS mode, and more. Since it can sit on top of a ZFS file-system, this ensures that all the data stored by ZFS, including in snapshots, is encrypted and will stay encrypted when performing functionality such as a `zfs send` to a remote backup server. This is a key feature of performing snapshots/replication with PC-BSD's new "Life-Preserver" utility. It enables you to safely transmit data to a remote machine (over SSH), without having to explicitly trust that the re-

mote system will take the necessary steps to re-encrypt your ZFS data, as in the case of GELI encryption.

Another key feature of PEFS is its inclusion of a PAM module for login decryption of a user's home-directory. So what does this look like? During the first time setup of PC-BSD, or when adding additional users, you will now be presented with a new option to "Encrypt user files".

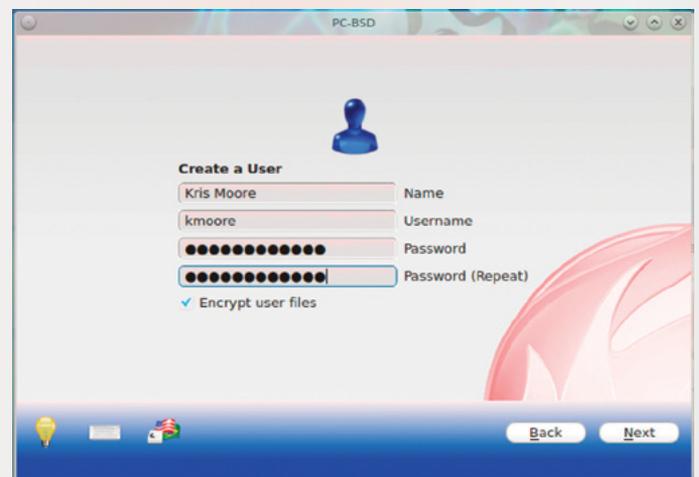


Figure 1. Setting up a new user with home-directory encryption

By enabling the "Encrypt user files" option, a new PEFS mount-point will be created on top of your home-directory, in this case `/usr/home/kmoore`. The encryption key for this

directory will be tied to your specified user password, so a password of good length and randomness is encouraged. When your user is not logged in, all the files in the home-directory will appear encrypted to root and to any other users who happen to log into the machine.

```
.W_6h0AJqhU7pQrX7+d_WDesdTdtF6N8WsgNiQIop8PKqN4X0rNqvsW
.ZZ1_SH_721nUa_j0NBkLReIob6p0Zv2qF
.amwUrcCQ9RLosRTNiBkPiN38qhUM71CW
.bggia iTTUk5sncLrWLaEsXdab6sQyfpFu04.jBS0qgtkrpywFfHep6xQ
.eprLTF9y5Bnob_SHb1zicWeCJcb1CCIF
.hlmIFpEUtdu3r.j5RuvA9SR8WE80SC53U
.j1kUy17ZSPXzagL+L4tWCox9tC2LcF8RsMd.j29BCf5PRKD3X5UBW5Q
.joqNwF4e7U7UFVcUX1K93a1nAbfWYjrNR2G0BAybFh8ZgyC1EZBdA
.mNUcfumgqz+oGGBCULk64kStXvRhk5vt
.pefs.db
.qTW3Uoxwm.jBL6dtq1kuZU0I1BpSR68Ka
.s+JE+AHWbxtNWHv0dqBa+8sZ.j0UannKsf
.s_0puhcUm6Uf iUZH10JcopuaDQnUzNyQ
.uYhxxv1UoUQ2p8+11rPA679TXKY.jwyfe2xCQHUqQw27sAKHUKa42aSA
.vTPAv1Gppqv533zP0MpLbv1EbwiZ.jN+u
.w07DN+WzaWyp0EY2SUyQkJwsyBZJSg0RDFso5UD5KEpZG+Fx_gEsg
.xNzEtvELmrn+c3sCi0mppmQm83dUcFeK
.xs0140osoh0WEwuu9iKhofB1RXTPhrZu1JcHzrL3L0cBQwNN_o25g
[root@titan] ~# ls /usr/home/kmoore/
```

Figure 2. Listing of a home-directory before login

After logging into the system as your user, your password will be automatically added to the PEFS key chain database and your encrypted directory will appear normally again.

```
[root@titan] ~# ls /usr/home/kmoore/
.cache                .vboxclient-seamless.pid
.config              .kde4                .windowlab
.cshrc               .login              .xprofile
.dbus                .login_conf         .xscreensaver
.dmrc                .mail_aliases       .xsession
.esd_auth            .mailrc             .xsession-errors
.fluxbox             .mounttray.settings .zshrc
.fvwm-crystal        .pefs.db            Desktop
.gconf               .profile            Documents
.gconfd              .pulse              Downloads
.gnome2              .pulse-cookie       GNUstep
.gtkrc-2.0           .rhosts             Images
.g3                  .screenrc           Music
.icons               .shrc               Videos
.icons               .vboxclient-clipboard.pid
.idesktoprc          .vboxclient-display.pid
[root@titan] ~#
```

Figure 3. Listing of the same home-directory after login

So how does this work behind the scenes? Let us take a look at the command-line usage for PEFS and its implementation on PC-BSD. As an example, we will walk through the creation of a new /secret directory on the system, where our user perhaps wants to keep more encrypted data with a different passphrase from his login. To get started, we will first create the new directory and initialize it with the pefs command.

```
[root@titan] ~# mkdir /secret
[root@titan] ~# pefs addchain -f -Z /secret
Enter parent key passphrase:
Reenter parent key passphrase:
[root@titan] ~#
```

Figure 4. Initial setup of the /secret file-system

Since this is a first-time initialization, and the PEFS directory is not yet mounted, we are going to use the

The BSD Certification Group Inc. (BSDCG) is a non-profit organization committed to creating and maintaining a global certification standard for system administration on BSD based operating systems.

WHAT CERTIFICATIONS ARE AVAILABLE?

BSDA: Entry-level certification suited for candidates with a general Unix background and at least six months of experience with BSD systems.

BDP: Advanced certification for senior system administrators with at least three years of experience on BSD systems. Successful BSDP candidates are able to demonstrate strong to expert skills in BSD Unix system administration.

WHERE CAN I GET CERTIFIED?

We're pleased to announce that after 7 months of negotiations and the work required to make the exam available in a computer based format, that the BSDA exam is now available at several hundred testing centers around the world. Paper based BSDA exams cost \$75 USD. Computer based BSDA exams cost \$150 USD. The price of the BDP exams are yet to be determined.

Payments are made through our registration website: <https://register.bsdcertification.org/register/payment>

WHERE CAN I GET MORE INFORMATION?

More information and links to our mailing lists, LinkedIn groups, and Facebook group are available at our website: <http://www.bsdcertification.org>

Registration for upcoming exam events is available at our registration website: <https://register.bsdcertification.org/register/get-a-bsdcg-id>

`-f` flag to skip the file-system checks. The `-z` flag is also used so we can create a new key-chain with only a single “parent” passphrase. This key-chain passphrase can then be verified at decryption time to ensure it was not mistyped.

PRO TIP

In addition to a single password, PEFS also supports creating parent / child keys in the keychain; giving the parent keys the ability to unlock the child keys, but not visa-versa. This allows you a method of creating security-levels in your encrypted data. For more information on this, take a look at the following wiki page: <https://wiki.freebsd.org/PEFS>.

With the initial chain created, if you take a look at the `/secret` directory, you will see that the `.pefs.db` key chain database file has been created, which is the only file that will need to exist underneath the PEFS mounted directory.

```
[root@titan] ~# ls /secret/
.pefs.db
[root@titan] ~# █
```

Figure 5. Verifying the contents of `/secret` before mounting PEFS

Now all that is left to do is to mount the PEFS file-system, and add the passphrase we just created. We can then look and still see the `.pefs.db` file, and see that the “pefs” file-system has been mounted successfully. This may be a good time to copy the `.pefs.db` file to a secure location, in case of accidental deletion. Note the usage of the `-c` flag on `pefs addkey`. This will enable a test ensuring that the passphrase entered is indeed the correct one from the key database.

```
[root@titan] ~# pefs mount /secret /secret
[root@titan] ~# pefs addkey -c /secret
Enter passphrase:
[root@titan] ~# ls /secret/
.pefs.db
[root@titan] ~# mount | grep '/secret'
/secret on /secret (pefs, local)
[root@titan] ~# █
```

Figure 6. Mounting PEFS and verifying the database file

With PEFS now mounted and ready, it is always a good practice to test your new setup before you begin using it with critical data. Next we will create a test file, verify it, remove our active encryption key, verify the encryption, and decrypt the directory again (Figure 7).

Lastly, we can setup the `/secret` directory to automatically have PEFS mounted at reboot. To enable this, you can add the directory you wish to re-mount with PEFS to

the `/var/db/pefs/auto_mounts` file. (You can safely create this file if it does not exist).

```
[root@titan] ~# echo "testing" > /secret/testing
[root@titan] ~# cat /secret/testing
testing
[root@titan] ~# pefs flushkeys /secret
[root@titan] ~# ls /secret
.Gk+LL3719lhsgM2e49bX52E110n7kC+6 .pefs.db
[root@titan] ~# pefs addkey -c /secret
Enter passphrase:
[root@titan] ~# ls /secret
.pefs.db testing
[root@titan] ~# █
```

Figure 7. Testing the encryption of the `/secret` directory

```
[root@titan] ~# echo "/secret" >> /var/db/pefs/auto_mounts
[root@titan] ~# █
```

Figure 8. Enabling auto-mounting of PEFS on `/secret`

References

- Article on PEFS usage / internals – <https://wiki.freebsd.org/PEFS>
- PEFS Sources – <https://github.com/glk/pefs>
- Original Article on PEFS by the author: Gleb Kurtsov – <http://glebkurtsov.blogspot.com/2009/09/pefs-crypto-primitives.html>
- PEFS in the FreeBSD ports tree – <http://www.freshports.org/sysutils/pefs-kmod/>

This method is preferred over a traditional entry in `/etc/fstab`, particularly, to ensure that your PEFS directories get mounted last and on top of other late-mounted file-systems, such as ZFS datasets. With this in place, you are now ready to begin using your encrypted directory normally. As a fail-safe, the directory is automatically mounted read-only, until you unlock it again via the `pefs addkey -c` command.

We have taken a quick look at PEFS: How it’s used on PC-BSD and how you can manually configure it via the command-line. For more information please read the excellent wiki article referenced below, or check out the source via GitHub. Traditional FreeBSD users may also find PEFS in the ports collection under `sysutils/pefs-kmod`.

KRIS MOORE

Kris Moore is the founder and lead developer of the PC-BSD project. He is also the co-host of the weekly BSDNow! video podcast with Allan Jude of ScaleEngine. When not at home programming, he travels around the world giving talks and tutorials on various BSD related topics at Linux and BSD conferences alike. He currently lives in Tennessee (USA) with his wife and five children and enjoys video gaming in his (very limited) spare time.

Faster. Better. Reliable.



Trusted by over 500 ISPs worldwide.

Hyper is the first multimedia cache fully developed in Brazil, by Taghos. With Hyper, ISPs can save on network bandwidth while increasing content-delivery speeds, resulting in end-customer satisfaction.

Features:

- 24x7x365 always-on support
- Active monitoring
- Automatic updates
- Appliance or license
- Easy deployment
- Configuration and reports via web interface



Remote Install
Using your hardware

Model	Traffic	RAM	Cache	SSD
T15	Up to 15 Mbps	8 GB	1x 1 TB	-
T50	Up to 50 Mbps	8 GB	2x 1 TB	-
T100	Up to 100 Mbps	8 GB	2x 1 TB	1x 160 GB
T150	Up to 150Mbps	16 GB	3x 2 TB	1x 160 GB
T300	Up to 300 Mbps	16 GB	5x 2 TB	1x 240 GB
T500	Up to 500 Mbps	32 GB	7x 2 TB	1x 480 GB
T1000	Up to 1 Gbps	64 GB	10x 1 TB	1x 480 GB
T2000	Up to 2 Gbps	96 GB	24x 1 TB	3x 480 GB
T3000	Up to 3 Gbps	128 GB	32x 1 TB	5x 480 GB

Visit us at www.taghos.com and start saving bandwidth today!

Intro to ZFS.

What is ZFS?

“The Z file system, originally developed by Sun™, is designed to use a pooled storage method in that space is only used as it is needed for data storage. It is also designed for maximum data integrity, supporting data snapshots, multiple copies, and data checksums. It uses a software data replication model, known as RAID-Z. RAID-Z provides redundancy similar to hardware RAID, but is designed to prevent data write corruption and to overcome some of the limitations of hardware RAID.”

ZFS is a modern 128-bit file system based on the copy-on-write model. It originates from the OpenSolaris project and first appeared in FreeBSD in 2008. ZFS has many innovative features including an integrated volume manager with mirroring and RAID capabilities, data checksumming and compression, writable snapshots that can be transferred between systems and many more. FreeBSD’s ZFS file system has been updated by merging improvements from the illumos project.

Current FreeBSD implementation of ZFS is ZFS Pool version 28. Here is the history of ZFS releases:

- 7.0+ – original ZFS import, ZFS v6; requires significant tuning for stable operation (no longer supported).
- 7.2 – still ZFS v6, improved memory handling, amd64 may need no memory tuning (no longer supported).
- 7.3+ – backport of new ZFS v13 code, similar to the 8.0 code
- 8.0 – new ZFS v13 code, lots of bug fixes – recommended over all past versions (no longer supported).
- 8.1+ – ZFS v14
- 8.2+ – ZFS v15
- 8.3+ – ZFS v28
- 9.0+ – ZFS v28

ZFS features :

- pooled storage (integrated volume manager)
- transactional semantics (copy-on-write)
- checksums and self-healing (scrub, resilver)
- scalability
- instant snapshots and clones
- dataset compression (lzjb)
- simplified delegable administration

Basic ZFS concepts

The ZFS file system uses two main objects: Pools and Datasets. A ZFS pool is a storage object consisting of virtual devices. These ‘vdevs’ can be:

- disk (partition, GEOM object, ...)
- file (experimental purposes)
- mirror (groups two or more vdevs)
- raidz, raidz2, raidz3 (single to triple parity RAID-Z)
- spare (pseudo-vdev for hot spares)
- log (separate ZIL device, may not be raidz)
- cache (L2 cache, may not be mirror or raidz)

Each ZFS pool contains ZFS datasets. ZFS dataset is a generic name for:

- file system (POSIX layer)
- volume (virtual block device)
- snapshot (read-only copy of file system or volume)
- clone (file system with initial contents of a snapshot)

For more information about this you can always check the handbook (http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/filesystems-zfs.html).

Requirements for this tutorial:

- FreeBSD production release (9.1)
- Around 512 MB of disk space (for simulating disks)
- At least 1 GB of RAM
- Root account

Purpose of this tutorial

The purpose of this tutorial is to explore some ZFS features in a safe way to grasp the power and flexibility of this file system. We will take a look at these basic functionalities:

- Create a ZFS pool.
- Create a ZFS mirror.
- Simulate a failure on a mirrored disk.
- Replace a disk.
- Adding disks to a mirrored zpool.
 - Check I/O on ZFS pools.

Creating Disks and Pools

To try some ZFS features, first we need to create pools. We will use files to simulate real disks so we can test things safely. I will use the `mkfile(8)` utility to create some files and use those as disks. `mkfile` creates one or more files that are suitable for use as NFS-mounted swap areas, or as local swap areas. The file is padded with zeros by default. The default size is in bytes, but it can be flagged as exabytes, petabytes, terabytes, gigabytes, megabytes, kilobytes, or blocks with the `e`, `p`, `t`, `g`, `m`, `k`, or `b` suffixes, respectively. Now let's create some disks! NOTE: If you don't have the `mkfile` utility, you just need to: Listing 1.

Here, I'm creating 4 disks of 128MB each as you can see in the `ls` output.

ZPools

All ZFS file systems live in a pool, so first we need to create a zpool. We can check pools with the `zpool(8)` command. Before creating new zpools, you should check for existing zpools to avoid confusing them with your tutorial pools. You can check what zpools exist with `zpool list`:

```
root@apollo:/array # zpool list
no pools available
```

Now let's create a zpool with `zpool create`:

```
root@apollo:/array # zpool create tutorial /array/disk00
```

Listing 1. Creating disks

```
root@apollo:/ # pkg_add -r mkfile

root@apollo:/ # mkdir array
root@apollo:/ # cd array
root@apollo:/array # mkfile 128m disk00 disk01 disk02
                        disk03
root@apollo:/array # ls -lrt
total 524416
-rw----- 1 root  wheel  134217728 Jul 29 21:51 disk00
-rw----- 1 root  wheel  134217728 Jul 29 21:51 disk01
-rw----- 1 root  wheel  134217728 Jul 29 21:51 disk02
-rw----- 1 root  wheel  134217728 Jul 29 21:51 disk03
```

Listing 2. Creating a new file on the pool

```
root@apollo:/ # mkfile 1m /tutorial/file1
root@apollo:/ # ls -lh /tutorial
total 1027
-rw----- 1 root  wheel   1.0M Jul 29 22:24 file1
root@apollo:/ #
```

Listing 3. Creating a new zpool

```
root@apollo:/ # zpool create example2 mirror /array/
                        disk00 /array/disk01
root@apollo:/ # zpool list
NAME      SIZE  ALLOC  FREE  CAP  DEDUP  HEALTH  ALTROOT
example2  123M   91K   123M   0%  1.00x  ONLINE  -
root@apollo:/ #
```

Listing 4. zpool status

```
root@apollo:/ # zpool status
pool: example2
state: ONLINE
scan: none requested
config:

NAME                STATE      READ WRITE CKSUM
example2             ONLINE    0    0    0
  mirror-0           ONLINE    0    0    0
    /array/disk00    ONLINE    0    0    0
    /array/disk01    ONLINE    0    0    0
```

List the current pools:

```
root@apollo:/array # zpool list
NAME      SIZE  ALLOC   FREE   CAP  DEDUP  HEALTH  ALTROOT
tutorial  123M   77K   123M    0%  1.00x  ONLINE  -
root@apollo:/array #
```

Now let's use the file system. Create a new file on the pool we just created (Listing 2). Here I have created a 1MB file on the newly created zpool.

Creating a ZFS mirror

A pool with only one disk doesn't offer any redundancy. Let's create a new zpool called "example2" using a couple of disks. We will use the keyword "mirror". As the name states, it will make a mirror using this pair of disks when we create the zpool (Listing 3).

We can check the status of our pools with the `zpool status` command (Listing 4).

Let's create a file again and check the status after that (I'll create a 32MB file):

```
mkfile 32m /example2/file
NAME      SIZE  ALLOC   FREE   CAP  DEDUP  HEALTH  ALTROOT
example2  123M  32.8M  90.2M   26%  1.00x  ONLINE  -
```

So now we have our data stored redundantly over the two disks.

Simulating a disk failure

Not everything is nice and calm. Sometimes bad things happen to good people, like a disk going bad at 3 a.m. Let's simulate a disk failure. For that I'll overwrite the first disk label with random data:

```
root@apollo:/ # dd if=/dev/random of=/array/disk01
      bs=1024 count=1
1+0 records in
1+0 records out
1024 bytes transferred in 0.029959 secs (34180 bytes/sec)
```

In case you don't know about the `dd(1)` command, here is what it does:

"The `dd` utility copies the standard input to the standard output. Input data is read and written in 512-byte blocks. If input reads are short, input from multiple reads are aggregated to form the output block. When finished, `dd` displays the number of complete and partial input and output blocks and truncated input records to the standard error output."

So I wrote a one-time block size of 1024 bytes from `/dev/random` to our `disk01`.

Listing 5. `scrub` command

```
root@apollo:/ # zpool scrub example2
root@apollo:/ # zpool status
pool: example2
state: DEGRADED
status: One or more devices could not be used because
       the label is missing or
       invalid. Sufficient replicas exist for the pool to
       continue
       functioning in a degraded state.
action: Replace the device using 'zpool replace'.
see: http://illumos.org/msg/ZFS-8000-4J
scan: scrub repaired 0 in 0h0m with 0 errors on Mon
       Jul 29 23:06:53 2013
config:

NAME                STATE      READ WRITE CKSUM
example2             DEGRADED   0     0     0
  mirror-0           DEGRADED   0     0     0
    /array/disk00    ONLINE    0     0     0
    2742290791000160517 UN
errors: No known data errors
```

Listing 6. `zpool attach`

```
root@apollo:/ # zpool attach example2 /array/disk00 /
                array/disk01
root@apollo:/ # zpool status example2
pool: example2
state: ONLINE
scan: resilvered 54.4M in 0h0m with 0 errors on Tue
       Jul 30 09:15:26 2013
config:
NAME                STATE      READ WRITE CKSUM
example2             ONLINE     0     0     0
  mirror-0           ONLINE     0     0     0
    /array/disk00    ONLINE    0     0     0
    /array/disk01    ONLINE    0     0     0
errors: No known data errors
```

Listing 7. Adding a disk to a Mirrored Zpool

```
root@apollo:/ # zpool add example2 mirror /array/disk02
                /array/disk03
root@apollo:/ # zpool list
NAME      SIZE  ALLOC   FREE   CAP  DEDUP  HEALTH  ALTROOT
example2  246M  54.4M  192M   22%  1.00x  ONLINE  -
root@apollo:/ #
```

Listing 8. zpool status example2

```

root@apollo:/ # zpool status example2
pool: example2
state: ONLINE
scan: resilvered 54.4M in 0h0m with 0 errors on Tue Jul 30
      09:15:26 2013

config:
NAME                STATE      READ WRITE CKSUM
example2            ONLINE    0    0    0
  mirror-0          ONLINE    0    0    0
    /array/disk00   ONLINE    0    0    0
    /array/disk01   ONLINE    0    0    0
  mirror-1          ONLINE    0    0    0
    /array/disk02   ONLINE    0    0    0
    /array/disk03   ONLINE    0    0    0

errors: No known data errors

```

Listing 9. zpool iostat -v

```

root@apollo:/ # zpool iostat -v
          capacity      operations      bandwidth
pool      alloc  free  read  write  read  write
-----
example2  54.4M  192M    0    0  1.44K   70
  mirror  54.4M  68.6M    2    3  193K  9.08K
    /array/disk00  -    -    0    0  2.31K  2.16K
    /array/disk01  -    -    0    5   214   210K
  mirror  14.5K  123M    0    0    0   307
    /array/disk02  -    -    0    0   268  26.2K
    /array/disk03  -    -    0    0   268  26.2K
-----

```

References

- <https://developer.apple.com/library/mac/documentation/Darwin/Reference/Manpages/man1/dd.1.html>
- http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/filesystems-zfs.html
- <http://manned.org/mkfile/96d2b7e1>

Now let's erase our file and create a new one to simulate a new disk:

```

root@apollo:/ # rm /array/disk01
root@apollo:/ # mkfile 128m /array/disk01

```

To attach another device, we specify an existing device in the mirror to attach it to with `zpool attach` (Listing 6). If you do type `zpool status` fast enough, after you attach the new disk, you will see a resilver (remirroring) in progress with `zpool status`. Once the resilver is complete, the pool is healthy again (you can also use `ls` to check the files are still there):

```

root@apollo:/ # ls /example2/
file file2

```

Adding a disk to a Mirrored ZPool

You can add disks to a zpool without taking it offline (Listing 7).

This happens almost instantly. Now `zpool status` returns that we have a pool composed of two mirrors (Listing 8).

ZFS automatically checks for errors when it reads/writes files; we can force a check with the `scrub` command (Listing 5). We messed up the disk, so it shows as UNAVAIL, but no errors are reported for the pool as a whole:

“Sufficient replicas exist for the pool to continue functioning in a degraded state.”

We still can read and write to the pool:

```

root@apollo:/ # ls -lrt /example2/
total 32779
-rw----- 1 root  wheel  33554432 Jul 29 22:41 file

```

Replacing a disk

Let's take out the bad disk from the pool using the `detach` command:

```

root@apollo:/ # zpool detach example2 /array/disk01

```

Checking I/O on ZPools

If we need to check IO on our pool, we have the `zpool iostat -v` (Listing 9).

All the data is currently written on the first mirror pair, as the second pair did not exist at the time the article was written.

That is all for this tutorial. Much more information on ZFS can be found in the following links :

- <http://docs.oracle.com/cd/E19253-01/819-5461/>
- http://www.solarisinternals.com/wiki/index.php/ZFS_Best_Practices_Guide
- <https://wiki.freebsd.org/ZFSTuningGuide>
- <http://wiki.illumos.org/display/illumos/illumos+Home>
- <http://manned.org/>

CARLOS ANTONIO NEIRA BUSTOS

FreeBSD on XenServer

In this article, we will learn the caveats in deploying FreeBSD over XenServer and the advantages over the traditional schema to improve administration, provisioning and delivery times.

What you will learn...

- Installation of FreeBSD in XenServer.
- Optimization of FreeBSD for XenServer.

What you should know...

- How to sync the source and build world and a custom kernel.
- XenServer management through XenCenter and console.
- FreeBSD installation procedure.

FreeBSD is an excellent platform; it is rock solid, easy to manage, easy to update, very well documented and has outstanding performance. All those reasons made me choose it over other Unix-like operating systems many years ago. However, configuration is time consuming and I wanted to reduce that in order to improve provisioning to replicate environments. After some research, I decided to use XenServer because it is used in many companies, plus it is free. This article shows how to install and customize FreeBSD over XenServer to achieve the maximum performance and improve administration, provisioning and delivery time, something critical in today's world of Cloud Computing.

FreeBSD Virtualization

Virtualization is a technique to implement resource separation and isolation of one physical computer into multiple environments called Virtual Machines. There are many techniques to implement this functionality with different benefits and their respective problems. This document is not a complete overview of FreeBSD virtualization or virtualization in general, but a review of some important information to understand where we can use XenServer, plus FreeBSD.

Virtual machines

This technique is designed to emulate hardware to run different operating systems on the same host computer. It

is the most used technique and we can test it using tools like VirtualBox, QEMU, and other virtual machine implementations. Here, each VM, or guest, runs without modifications because the virtualization software emulates the hardware and the guest is not aware that it is not running on real hardware. This is an unnecessarily heavy solution, but it's the best option if we want operating systems like Windows to do our tax report from BSD platforms.

OS level virtualization or Jails

This technique is a layer on the Operating System designed to isolate environments inside the main system. You have a base system with many environments working on the same platform version, for example, FreeBSD 9.1. Each virtual machine or jail has its own IP address or addresses with processes, system accounts, applications and files exclusive for that environment. The advantage of this technique is the low overhead of each environment because they share the same resources such as memory, CPUs, disks and network, while maintaining full isolation. On the other hand, the disadvantages are the same resource sharing since when we have a jail with high resource consumption, this can affect the performance of another jail or the host in general.

Paravirtualization

This technique uses an operating system with a small footprint called a *hypervisor*, specifically designed to handle

resource assignation and interruptions between guests. The guest operating systems (or VM) must be adapted to be paravirtualization-aware to take full advantage of it.

The advantage of this technique is that you can run different operating systems on the same computer without the overhead of virtual machines. XenServer fits in this kind of virtualization and FreeBSD is paravirtualizations-aware through the XENHVM kernel.

As a side note, the BSD community is working on a hypervisor called *bhyve*, The BSD Hypervisor. It is designed to support different versions of FreeBSD and it has been claimed to run CentOS at BSDCon 2013, so it looks like a promising solution.

Virtual Machine Setup

The first step, even before we start with the installation of FreeBSD, is to download the ISO image with the installer and save it in the NFS ISO repository in XenServer. For this article, we will be using the FreeBSD 9.1 installer because 9.2 has not been released yet.

To simplify the process, we will be using a Windows application called XenCenter under a virtualized Windows and we can begin the installation as you see in Figure 1.

Because XenServer does not have a specific FreeBSD template, we need to follow the “Other install media” option and advance to the next screen. We are not providing any screenshots for the installation process because it is really easy and we are focused on showing you how to make FreeBSD work.

In the next screens, you can choose the amount of RAM and vCPUs assigned to the VM. It is important to assign plenty of RAM because we will build a custom kernel (and world). Once we have finished, we can reassign it to free memory for other Virtual Machines.

Next, we need to create a custom disk and choose the right network interface. It is important to create a disk with enough space to build world and the custom kernel although it is possible to increase the size of the disk after the initial setup, but it requires additional steps. Later, we will explain how to do it.

Finally, we will see a confirmation screen, as shown in Figure 2, with the summary of options.

If we leave the option “Start the new VM automatically” checked, the FreeBSD installer will run immediately as we can see in Figure 3.

And that is all for the VM setup. Next, we will install FreeBSD using the XenCenter Console.

FreeBSD Setup

Here we will do a common FreeBSD installation without customization options like ZFS or other customized options.

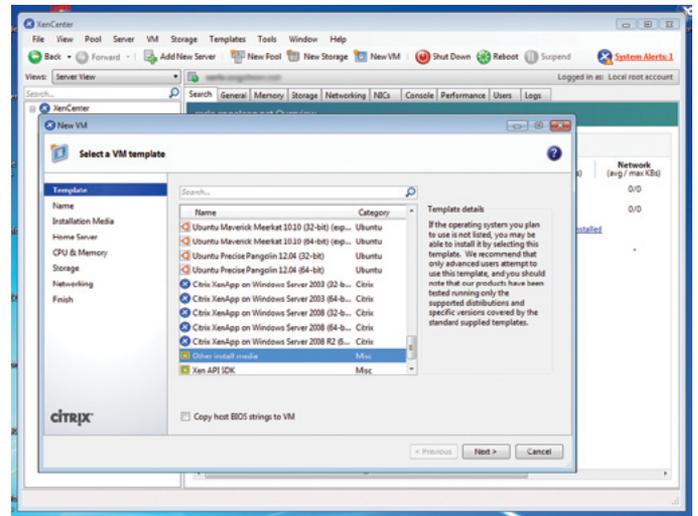


Figure 1. VM's template selection

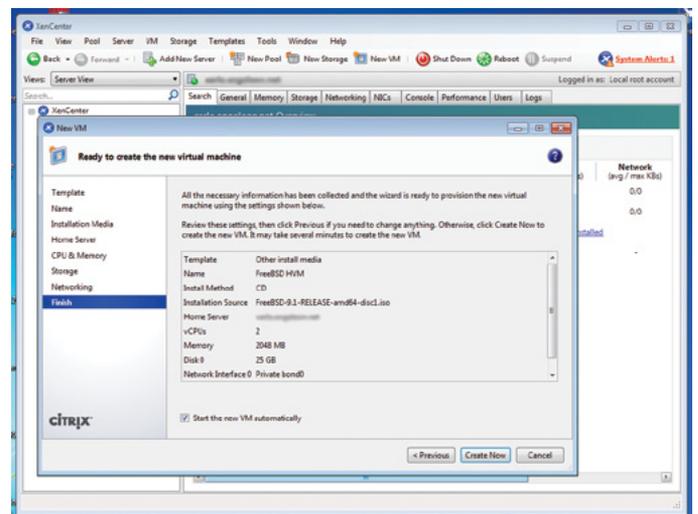


Figure 2. VM's summary options

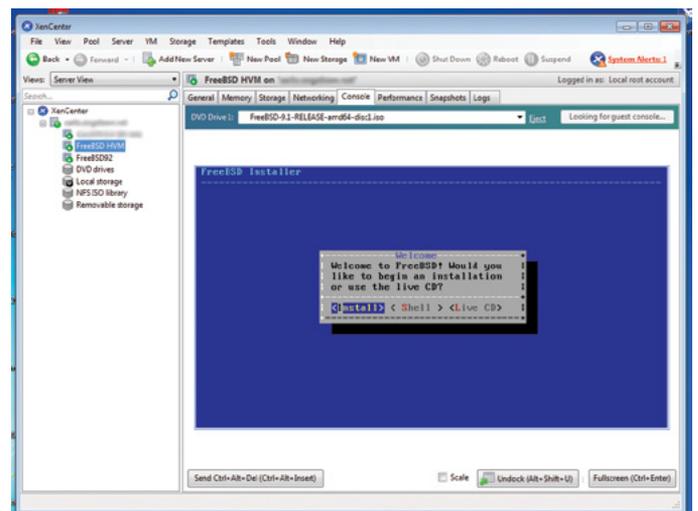


Figure 3. FreeBSD Installer

We can leave out the “src” and “ports” components from the install because later, we will download the latest versions.

The only relevant thing here is when we will create the partitions because, as mentioned previously, if disk resizing is desired, we will need to leave root, or the growing partition at the end of the list. In that way, we can resize the disk in XenServer and the UFS partition (using growfs) without problems.

You can check the setup used for this case in Figure 4.

After finishing the installation, the first thing we need to do is shut down the VM and remove the CD/DVD drive from the VM. This is not required yet but it is a must when we build the custom HVM kernel otherwise we will receive the message “run_interrupt_driven_hooks: still waiting after X seconds for xenbusb_nop_confighook_cb” continuously, and it will never boot. To do this, first login to the XenServer console and run the commands on Listing 1.

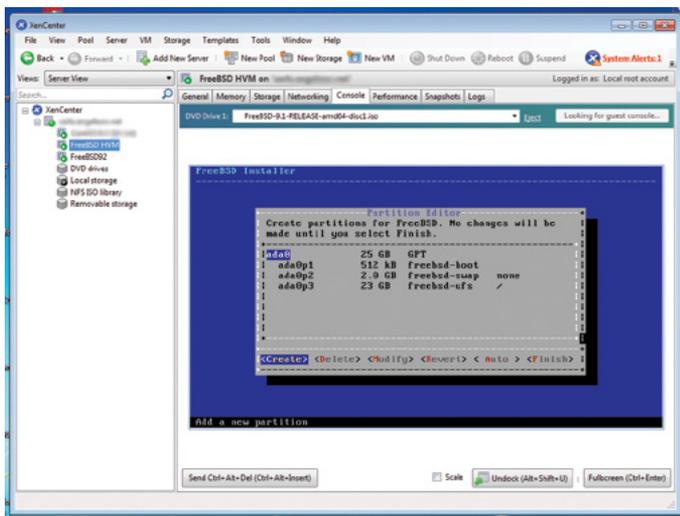


Figure 4. FreeBSD partitioning

Remember to run the `xe vbd-destroy` command with the VM shutdown otherwise you will get the message “You attempted an operation that was not allowed.”

We can now turn the VM on and log in as root. The next step is to install subversion to do the checkout of the source, build the “world” and *XENHVM* kernel as seen in Listing 2. After compilation, we must change the final settings *before restarting*. The first one is updating the network interface in *rc.conf* from “ifconfig_reX” to “ifconfig_xnX” because an HVM kernel renames the interface to xnX. If we wish to do offloading, we can add the flags “-tx-csum -rx-csum -lro -tso” also.

I have read on the freebsd-xen mailing list that if you are using *pf*, you must set `net.inet.tcp.tso=0`.

Also, many websites write about the need to update the *fstab* and change all the references from “adaX” to “adX”, but in this version of XenServer (6.2) and FreeBSD (9.x), it was not required. You can leave your *fstab* as it is. Check Listing 3 for further information.

Now we can restart the VM using `shutdown -r now` and we will boot using the new HVM kernel and the latest version of FreeBSD. At the moment of this writing, it is 9.2-PRERELEASE.

Maybe you’re wondering why I’m going through all this trouble when a vanilla FreeBSD will work great on its own? The reason is performance and support. 64-bit Intel/AMD (amd64) kernels are only supported using HVM and the performance gain is through PV drivers, supported in HVM configurations.

Virtualization advantages

Now after having a FreeBSD VM completely configured, we must comment on the advantages over a traditional setup. The more notable are variety of versions, securi-

Listing 1. Destroy CD/DVD from the VM

```
# find the uuid for the VM
[root@xenserver ~]# xe vm-list name-label='FreeBSD HVM'
uuid ( RO)           : 8ff002a0-3f79-a1b4-96fa-5e213143ea25
  name-label ( RW): FreeBSD HVM
  power-state ( RO): running

# get the uuid for the CD/DVD
[root@xenserver ~]# xe vbd-list vm-uuid=8ff002a0-3f79-a1b4-96fa-5e213143ea25 device=hdd params=uuid
uuid ( RO)          : 3419ad05-7502-5344-1458-568b945e9646

# destroy the CD/DVD
[root@xenserver ~]# xe vbd-destroy uuid=3419ad05-7502-5344-1458-568b945e9646
[root@xenserver ~]#
```

ty and isolation, backup and finally, provisioning. By variety of versions, I mean different versions of FreeBSD like 7, 8, 9 or CURRENT, and other variants of BSD like OpenBSD and NetBSD – something impossible to do using only jails. Next, although we have jails and chroot to do security and isolation and they have plenty of benefits, this kind of isolation allows us to restrict the amount of resources, such as memory and CPU by VM, not just by user or group of users.

Backup is more robust to do because we receive the benefit of full VM snapshots. It is easier to back up the entire VM before doing something risky and if that test does not work, return to the previous state.

I know we can do something similar using UFS or ZFS snapshots. But the problem is that they are still inside of the operating system/filesystem while VM's snapshots are outside of the OS, giving us other advantages such as the possibility of creating new VMs using them.

The advantages for provisioning are enormous because with one snapshot from a guest, we can create images and templates with a different stack of applications and services such as web, mail, file servers, etc., and then use those to create environments for development, testing, quality assurance and production – all in a matter of minutes and not hours/weeks as in a traditional deployment.

Another advantage in using XenServer is the integration with Cloud Stack to give customers a self-service portal.

On the web

- <http://www.freebsd.org/> – FreeBSD
- <http://www.xenserver.org/> – OpenSource XenServer
- <http://bhyve.org> – The BSD Hypervisor
- <http://home.uncon.net/dokuwiki/doku.php?id=xenserver:freebsd> – FreeBSD Tweaking

Summary

This article was an overview of how to run FreeBSD on XenServer to make it more dynamic and easy-to-manage, more efficiently, with more control, isolation and better resource assignment, all to improve our services and customers' experience.

ANGEL LEON

Angel Leon has been working on high demand websites for the Mexican Government and has more than nine years of experience with BSD systems. Currently he works as a consultant, helping companies construct robust platforms for high availability and high performance. He writes about technical topics in his free time for the portal, <http://sys-admin.org.mx> and can be reached at <http://nixnetworks.mx/contact>.

Listing 2. Build the custom kernel and world

```

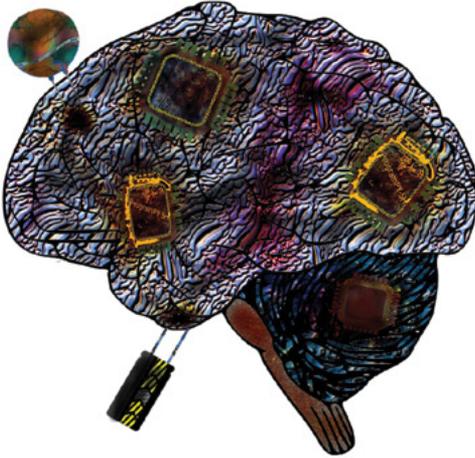
root@freebsdhvmdemo:/root # pkg_add -r subversion
Fetching ftp://ftp.freebsd.org/pub/FreeBSD/ports/amd64/packages-9.1-release/Latest/subversion.tbz... Done.
Fetching ftp://ftp.freebsd.org/pub/FreeBSD/ports/amd64/packages-9.1-release/All/expat-2.0.1_2.tbz... Done.
Fetching ftp://ftp.freebsd.org/pub/FreeBSD/ports/amd64/packages-9.1-release/All/pkgconf-0.8.9.tbz... Done.
...
root@freebsdhvmdemo:/root # rehash
root@freebsdhvmdemo:/root # svn checkout http://svn0.us-west.FreeBSD.org/base/stable/9 /usr/src
A   /usr/src/Makefile
A   /usr/src/sys
A   /usr/src/sys/boot
A   /usr/src/sys/boot/Makefile.inc
A   /usr/src/sys/boot/Makefile.amd64
A   /usr/src/sys/boot/common
...
root@freebsdhvmdemo:/root # make buildworld
...
root@freebsdhvmdemo:/root # make buildkernel KERNCONF=XENHVM
...
root@freebsdhvmdemo:/root # make installkernel KERNCONF=XENHVM
...
root@freebsdhvmdemo:/root # mergemaster -p
...
root@freebsdhvmdemo:/root # make installworld
...
root@freebsdhvmdemo:/root # mergemaster
...
root@freebsdhvmdemo:/root # make delete-old
...

```

Listing 3. fstab

#	Device	Mountpoint	FStype	Options	Dump	Pass#
	/dev/ada0p2	none	swap	sw 0 0		
	/dev/ada0p3	/	ufs	rw 1 1		

Current Projects by Devyn Collier Johnson **<DevynCJohnson@Gmail.com>**



Neobot is one of several Betabots. All Betabots are advanced chatbots that use the Pysh engine and read Xaiml files. The Betabots, Pysh, and Xaiml are still developing technologies made by Devyn Collier Johnson. Neobot and the Xaiml specification can be found here <https://launchpad.net/neobot>

I make many wallpapers for free.
Come check them out <http://gnome-look.org/usermanager/search.php?username=DevynCJohnson>



I write articles for Linux.org. Come check out the place and enjoy the site. Linux.org offers forums and tutorials as well as informative articles.

Want me as your writer? Email me your name, website/company, email address, a list of desired articles (topic, article size, and so on), and payment method/amount. Feel free to ask me further questions. To learn more about me, go to this site (<https://launchpad.net/~devyncjohnson-d>).

I am also the beta-tester for Wallch and SuperTux.



NET OPEN SERVICES IS AN APPLICATION HOSTING COMPANY FOCUSED ON OPEN SOURCE APPLICATIONS MANAGEMENT IN HIGH AVAILABILITY ENVIRONMENT.

NET OPEN SERVICES IS PROUD TO PROVIDE A HIGH QUALITY SERVICE TO OUR CUSTOMERS SINCE 10 YEARS.

OUR EXPERTISE INCLUDES:

- CLOUD COMPUTING, PUBLIC, PRIVATE AND HYBRID CLOUD MANAGEMENT (OPENSTACK, CLOUDSTACK, RED HAT ENTERPRISE VIRTUALIZATION)
- REMOTE MONITORING AND MANAGEMENT 24/7
- NETWORKING AND SECURITY (OPEN BSD, IP TABLE, CHECKPOINT, CISCO,...)
- OS AND APPLICATION MANAGEMENT (FREE BSD, OPEN BSD, SOLARIS, UNIX, LINUX, AIX, MS WINDOWS)
- DATABASE MANAGEMENT (ORACLE, MYSQL, CASSANDRA, NOSQL, MS SQL, SYBASE...)
- MANAGED HOSTING IN CARRIER CLASS DATA CENTERS
- DISASTER RECOVERY



WE PROVIDE SERVICES IN EVERY STEP OF THE PROJECT LIFE, DESIGN, DEPLOYMENT, MANAGEMENT AND EVOLUTIONS. NETOPENSERVICES TEAM INCLUDES EXPERIENCED LEADERS AND ENGINEERS IN THE INTERNET SERVER INDUSTRY.

OUR TEAM HAS 15 YEARS OF EXPERIENCE IN DEVELOPING INTERNET INFRASTRUCTURE-GRADE SOLUTIONS AND PROVISIONING INTERNET DATACENTERS AND GLOBAL SERVICE NETWORKS TOGETHER.

WE OFFER EXCEPTIONAL HARDWARE SUPPORT AS SOFTWARE SUPPORT ON UNIX/LINUX AND OPEN SOURCE APPLICATION. NETOPENSERVICES DELIVERS THESE CUSTOM-BUILT LINUX AND UNIX SERVERS, AS WELL AS PRECONFIGURED SERVERS AND SCALABLE STORAGE SOLUTIONS, TO OUR CUSTOMERS. WE ALSO OFFER CUSTOM DEVELOPMENT AND ADVANCED-LEVEL UNIX/LINUX CONSULTING SOLUTIONS.



Headquarters:
San Jose, CA



855.GREP.4.IX | Contact Us

99% Compatibility

online now...

IXSYSTEMS AND YOU ARE
THE PERFECT MATCH



SHARED INTERESTS

- Enterprise Storage Solutions
- Personalized Customer Service
- Bold New Information Technology

I'm a

In

Looking for

- A Technology Partner
- More Technical Experience
- New Business Opportunities

Visit Today!



iXsystems

Technology Partner Seeking
Resellers/Integrators for
TrueNAS™ Storage Appliance



WWW.IXSYSTEMS.COM/PERFECTMATCH

