SAINT®

SAINT v7.0 Now Available
www.saintcorporation.com

Integrated
Vulnerability Scanning
and Penetration Testing

BSD MAGAZINE

FOR NOVICE AND ADVANCED USERS

BSD AS A DESKTOP

Spam Control with a stock OpenBSD install

Build Your Own FreeBSD Update Server

Keeping FreeBSD Applications Up-To-Date

Using OpenBSD and PF as a Virtual Firewall for Windows

BSD Goes to the Office: Can BSD compete in a real life consulting workplace?

EXCLUSIVELY

▶ Interview: BSD Live Desktops

Vol.3 No.3
Issue 3/2010(9)

iX systems™

800-820-BSDi
http://www.iXsystems.com
Enterprise Servers for Open Source

✓ Increased Performance ✓ Impressive Energy Savings

Energy Efficient, Powerful Performance

With dual Intel® Xeon® 5500 series Quad-Core or Dual-Core processors and up to 144GB of DDR3 memory, the iX-Athena is designed for small businesses seeking a high performance computing solution.

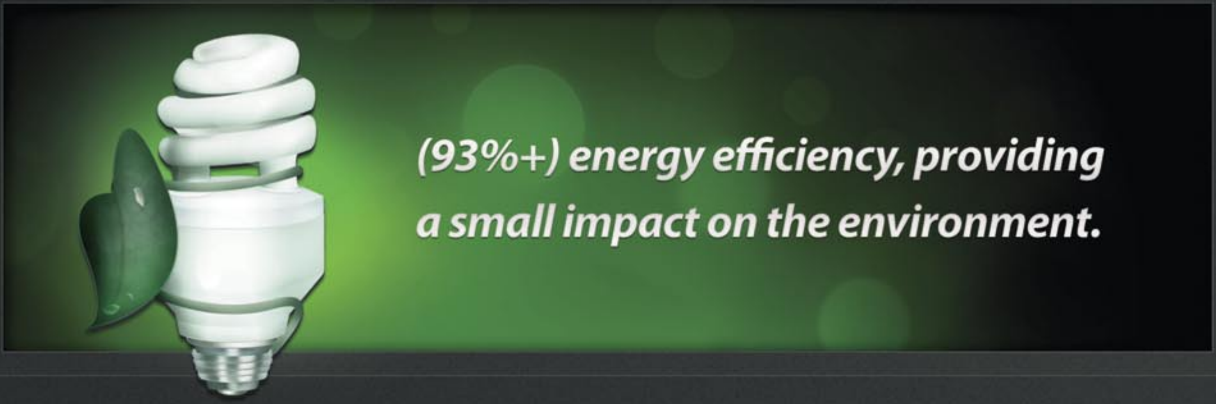
iX-Athena

Notable features include:

- Dual 64-Bit Socket 1366 Quad-Core or Dual-Core, Intel® Xeon® Processor 5500 Series
- Eight 3.5" Hot-swap SAS/SATA HDDs in a 4U/Tower Configuration (Optional 4U Rackmount Rail Kit Available)
- Dual Intel® 5520 Chipsets with Quick-Path Interconnect (QPI) up to 6.4 GT/s
- Up to 144GB DDR3 1333/1066/800MHz ECC Registered DIMM/24GB Unbuffered DIMM (18 DIMM Slots)
- Two (x16) PCI-E 2.0 slots, Four (x8) PCI-E 2.0 slots (1 in x 16 slot), and One (x4) PCI-E slot (in x8 slot)
- Intel® 82576 Dual-Port Gigabit Ethernet Controller
- Matrox G200eW Graphics Support
- Integrated IPMI 2.0 with Dedicated LAN
- Realtek ALC888 7.1 HD audio
- Two 5,000 RPM Hot-swap Cooling Fans
- Two 5,000 RPM Hot-swap Rear Exhaust Fans
- 1400W Redundant High Efficiency Power Supply (Gold Level 93%+ power efficiency)

Front View

Back View


(93%+) energy efficiency, providing a small impact on the environment.

iXsystems Presents the iX-Athena Workstation

The iX-Athena showcases amazing computing performance and energy efficiency, while keeping noise levels to a minimum.

The iX-Athena delivers the most powerful performance available on the market today. Dual Intel® Xeon® 5500 series Quad-Core or Dual-Core processors boost performance for specific workloads by increasing processor frequency. Next-generation Intel® Virtualization Technology enhances performance in virtualized environments by up to 2.1x with new hardware-assist capabilities. Up to 144GB of DDR3 memory with eighteen DIMM sockets supports higher performance for data-intensive applications and makes it easy for the iX-Athena to handle any workload.

In terms of energy efficiency, the iX-Athena also leads the pack. The automated low-power states of the Intel® Xeon® 5500 series processors intelligently save power during low-use periods and increase performance when the system requires it. The iX-Athena also features an FCC Class B certified power supply with gold level (93%+) energy efficiency to provide 1400W of power and minimize impact on the environment.

The Super-Quiet operation of the iX-Athena allows users to spend less time distracted by a loud machine, and more time focusing on its powerful computing capabilities. At normal operation levels, the iX-Athena workstation's 5,000 RPM cooling and exhaust fans perform at a hushed 38 decibels to make this an ideal machine for any office or lab environment.

With eight 3.5" hot-swappable SAS/SATA hard drive bays, the iX-Athena also offers ample storage for all conceivable technical computing and graphics applications. The iX-Athena even includes four dedicated power connectors for high-end graphics cards, all contained in a stylishly sleek, high-end quality, dark gray chassis.

To order today call:
1-800-820-BSDi

For more information about the iX-Athena visit:
http://www.iXsystems.com/Athena

iX systems™

intel® Xeon® inside™
Powerful. Intelligent.

Dear Readers!

BSD is already becoming international magazine. People all over the world have an access to our magazine and download it. We are happy that our work is so appreciated and BSD magazine popularity is growing!.

First of all I wanted to thank you for you letters of support, they mean really a lot to us and help constantly to improve! All our authors worked hard to make their articles interesting and useful. I really hope you will like this issue as much as the previous.

This month topic is "BSD as a desktop". Why this topic?

We thought that some of you still might have doubts on choosing OS, so this issue surely will help you to learn more about BSD as a desktop and help to make a decision.

But those of you who already use BSD should not close the magazine after reading my previous statement, because you could loose a lot. =)

Please feel free to contact us, we are open to critics, not only to new ideas and suggestions.

Your feedback is very important to us.

Olga Kartseva
Editor in Chief

MAGAZINE BSD

Editor in Chief:
Olga Kartseva
olga.kartseva@software.com.pl

Contributing:
Jan Stedehouder, Rob Somerville, Marko Milenovic, Petr Topiarz, Paul McMath, Eric Vintimilla, Matthias Pfeifer, Theodore Tereshchenko, Mikel King, Machtelt Garrels, Jesse Smith

Special thanks to:
Marko Milenovic, Worth Bishop and Mike Bybee

Art Director:
Agnieszka Marchocka

DTP:
Ireneusz Pogroszewski

Senior Consultant/Publisher:
Paweł Marciniak *pawel@software.com.pl*

National Sales Manager:
Ewa Łozowicka
ewa.lozowicka@software.com.pl

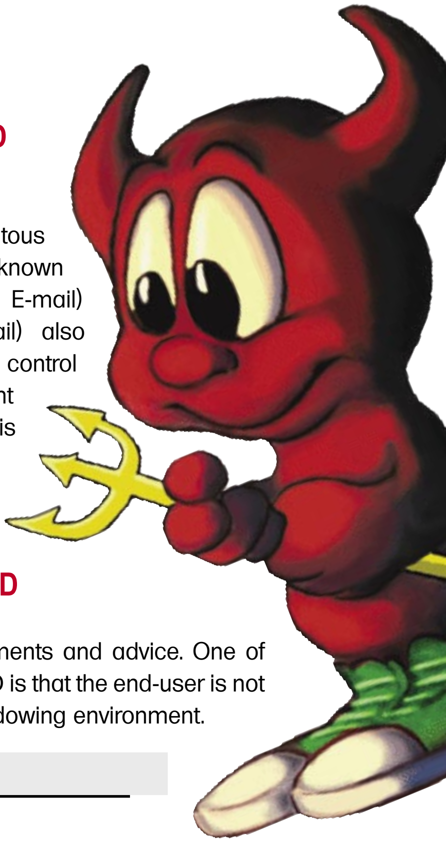
Marketing Director:
Ewa Łozowicka
ewa.lozowicka@software.com.pl

Executive Ad Consultant:
Karolina Lesińska
karolina.lesinska@bsdmag.org

Advertising Sales:
Olga Kartseva
olga.kartseva@software.com.pl

Publisher :
Software Press Sp. z o.o. SK
ul. Bokserska 1, 02-682 Warszawa
Poland
worldwide publishing
tel: 1 917 338 36 31
www.bsdmag.org

Software Press 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*



Build Your Own FreeBSD Update Server

Jason Helfman

Experienced users or administrators responsible for several machines, or environments, know the difficult demands and challenges of maintaining such an infrastructure.

Running a FreeBSD Update Server makes it easier to deploy security and software patches to selected test machines before rolling them out to production. It also means a number of systems can be updated from local network rather than a much slower Internet connection. This article outlines the steps involved in creating an internal FreeBSD Update Server.

Prerequisites

To build an Internal FreeBSD Update Server you will need the following.

- A running FreeBSD system.
- A user account with at least 4Gigs of available space. This will allow for the creation of at least updates for 7.1 and 7.2. Beyond this space requirements will need to be considered.
- An `ssh(1)` account on a remote machine to upload distributed updates. See the man page here: *http://www.freebsd.org/cgi/man.cgi?query=ssh&sektion=1*.
- An Apache, *http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/network-apache.html*, web server, with over half of the the space required for the build. For instance, my builds total 4G, and the webserver space needed to distribute updates is 2.6G.
- Basic knowledge of shell scripting with Bourne shell, `sh(1)`. See the man page here: *http://www.freebsd.org/cgi/man.cgi?query=sh&sektion=1.*

Configuration: Installation & Setup

Download freebsd-update-server software at *http://www.freebsd.org/cgi/cvsweb.cgi/projects/freebsd-update-server/*. A tarball may be downloaded, or use `csup(1)` and the projects-all collection. See the man page here for csup: *http://www.freebsd.org/cgi/man.cgi?query=csup&sektion=1*.

Update `scripts/build.conf` appropriately. It is sourced during build operations.

Here is the default `build.conf.default`, which should be modified (Listing 1).

Parameters for consideration would be:

- FTP – This is where the subroutine `fetchiso()` declared in scripts/build.subr will contact the configured source for downloading the FreeBSD ISO. This can be configured to be an http address, as well. For our purposes, ISO's are on the same server as our internal http server that will be serving updates. The software has been configured to look in that location. For this setup, we have to alter the routine to fetch the ISO. By copying the source `build.subr` to `scripts/RELEASE/ARCHITECTURE/build.subr` this file will be sourced instead of the released source for `build.subr`.
- BUILDHOSTNAME – Host where software will build. Coincidentally, this information will be displayed on updated systems when issuing: `% uname -v`
- SSHKEY – Key for uploading to update server where clients will fetch patches or upgrades. A key pair is created by

This article describes building an internal FreeBSD Update Server. The freebsd-update-server software, located at *http://www.freebsd.org/cgi/cvsweb.cgi/projects/freebsd-update-server/*, is written by Colin Percival *cperciva@FreeBSD.org*; Security Officer of FreeBSD. If you thought it was fun to update your system against an Official Update Server, just wait until you have an updated system from your very own FreeBSD Update Server.

executing `ssh-keygen -t dsa`. Altering this parameter is not necessary, as standard password authentication through ssh will suffice if configured properly. `ssh-keygen(1)` has more detailed information in creating a key pair. The man page will have more information, and it can be found here: *http://www.freebsd.org/cgi/man.cgi?query=ssh-keygen&sektion=1*.

- MASTERACCT – Account that files are uploaded to on remote system.
- MASTERDIR – Directory where files are uploaded to on remote system.

Now that build directives are set, the installation files are configured for a build. For this example, we will use RELEASE-7.2 under amd64 architecture. Configuration files for i386 architecture are available with downloaded source.

Create the build environment directory under `scripts/RELEASE-7.2/amd64`.

```
 % mkdir -p /usr/local/freebsd-update-
server/scripts/RELEASE-7.2/amd64
```

This is the `build.conf` file that should be placed in the directory that was created in the previous step (see Listing 2).

Note

To generate the *End of Life* number for `build.conf`, refer to the *Estimated EOL* posted on the FreeBSD Security Website at *http://www.freebsd.org/security/security.html*.

Based on this date, you can issue date `-j -f '%Y%m%d-%H%M%S' '20090401-000000' +%s`, and substitute actual date parameters for those stated by FreeBSD.

The SHA256 hash key for the desired release, is published within the respective release announcement found at *http://www.freebsd.org/releases/*.

Building Update Code

The first step is to run `scripts/make.sh`. This will build some binaries, create directories, and generate an RSA signing key used for approving builds. In this step, a passphrase will have to be supplied for the final creation of the signing key (see Listing 3).

Listing 1. Installation and Setup 1st step

```
 # $FreeBSD: projects/freebsd-update-server/scripts/build.conf,v 1.1 2006/08/
31 07:48:40 cperciva Exp $
 # Main configuration file for FreeBSD Update builds.  The
 # release-specific configuration data is lower down in
 # the scripts tree.
 # Location from which to fetch releases
 export FTP=ftp://ftp2.freebsd.org/pub/FreeBSD/releases
 # Host platform
 export HOSTPLATFORM='uname -m'
 # Host name to use inside jails
 export BUILDHOSTNAME=${HOSTPLATFORM}-builder.daemonology.net
 # Location of SSH key
 export SSHKEY=/root/.ssh/id_dsa
 # SSH account into which files are uploaded
 MASTERACCT=builder@wadham.daemonology.net
 # Directory into which files are uploaded
 MASTERDIR=update-master.freebsd.org
```

Listing 2. Installation and Setup 2nd step

```
 # SHA256 hash of RELEASE disc1.iso image.
 export RELH=1ea1f6f652d7c5f5eab7ef9f8edbed50cb664b08ed761850f95f48e86cc71ef5
 # Components of the world, source, and kernels
 export WORLDPARTS="base catpages dict doc games info manpages proflibs lib32"
 export SOURCEPARTS="base bin contrib crypto etc games gnu include krb5  \
                 lib libexec release rescue sbin secure share sys tools  \
                 ubin usbin cddl"
 export KERNELPARTS="generic"
 # EOL date
 export EOL=1275289200
```

Listing 3. Building Update Code. Final creation of a signing key

```
# sudo sh scripts/make.sh
 cc -O2 -fno-strict-aliasing -pipe   findstamps.c  -o findstamps
 findstamps.c: In function 'usage':
 findstamps.c:45: warning: incompatible implicit declaration of built-in
function 'exit'
 cc -O2 -fno-strict-aliasing -pipe   unstamp.c  -o unstamp
 install findstamps ../bin
 install unstamp ../bin
 rm -f findstamps unstamp
 Generating RSA private key, 4096 bit long modulus
 ......................................................................++
 ...................++
 e is 65537 (0x10001)
 Public key fingerprint:
 27ef53e48dc869eea6c3136091cc6ab8589f967559824779e855d58a2294de9e
 Encrypting signing key for root
 enter aes-256-cbc encryption password:
 Verifying - enter aes-256-cbc encryption password:
```

Note

Take down the generated *KeyPrint*; this value is entered into `/etc/freebsd-update.conf` for binary updates. At this point, we are ready to stage a build.

```
# cd /usr/local/freebsd-update-server
# sh scripts/init.sh amd64 RELEASE-7.2
```

What follows is sample of an initial build run (see Listing 4).

Note

Then the build of the world is performed again, with world patches. A more detailed explanation may be found in `scripts/build.subr`.

Note

And then the build completes... Approve the build if everything looks ok. More information on determining if things are ok can be found in the distributed source file named USAGE.

Execute `scripts/approve.sh`, as directed. This will sign the release, and move components into a staging area suitable for uploading. It is important to make sure that your key is mounted during this process. A simple df will show if it is mounted. If not mounted, mount the key with the passphrase supplied when creating it earlier (see Listing 7).

```
# cd /usr/local/freebsd-update-server
# sh scripts/mountkey.sh
```

Listing 4. Building Update Code. Initial build run

```
# sh scripts/init.sh amd64 7.2-RELEASE
Mon Aug 24 16:04:36 PDT 2009 Starting fetch for FreeBSD/amd64 7.2-RELEASE
/usr/local/freebsd-update-server/work/7.2-RELE100% of  588 MB  359 kBps 00m00s
Mon Aug 24 16:32:38 PDT 2009 Verifying disc1 hash for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 16:32:44 PDT 2009 Extracting components for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 16:34:05 PDT 2009 Constructing world+src image for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 16:35:57 PDT 2009 Extracting world+src for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 23:36:24 UTC 2009 Building world for FreeBSD/amd64 7.2-RELEASE
Tue Aug 25 00:31:29 UTC 2009 Distributing world for FreeBSD/amd64 7.2-RELEASE
Tue Aug 25 00:32:36 UTC 2009 Building and distributing kernels for FreeBSD/amd64 7.2-RELEASE
Tue Aug 25 00:44:44 UTC 2009 Constructing world components for FreeBSD/amd64 7.2-RELEASE
Tue Aug 25 00:44:56 UTC 2009 Distributing source for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 17:46:18 PDT 2009 Moving components into staging area for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 17:46:33 PDT 2009 Identifying extra documentation for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 17:47:13 PDT 2009 Extracting extra docs for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 17:47:18 PDT 2009 Indexing release for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 17:50:44 PDT 2009 Indexing world0 for FreeBSD/amd64 7.2-RELEASE

Files built but not released:
Files released but not built:
Files which differ by more than contents:
Files which differ between release and build:
kernel|generic|/GENERIC/hptrr.ko
kernel|generic|/GENERIC/kernel
src|sys|/sys/conf/newvers.sh
world|base|/boot/loader
world|base|/boot/pxeboot
world|base|/etc/mail/freebsd.cf
world|base|/etc/mail/freebsd.submit.cf
world|base|/etc/mail/sendmail.cf
world|base|/etc/mail/submit.cf
world|base|/lib/libcrypto.so.5
world|base|/usr/bin/ntpq
world|base|/usr/lib/libalias.a
world|base|/usr/lib/libalias_cuseeme.a
world|base|/usr/lib/libalias_dummy.a
world|base|/usr/lib/libalias_ftp.a
...
...
```

After completing the approval process, you may proceed with the upload.

```
# cd /usr/local/freebsd-update-server
# sh scripts/upload.sh amd64 RELEASE-7.2
```

The uploaded files will need to be in the DocumentRoot of the webserver in order for updates to be distributed. For further explanation, please refer to the Configuration section of the Apache documentation.

Note

Updates for the current release of the FreeBSD system you are updating, and what you want to upgrade to need to be built in order for FreeBSD Update Server to work properly. This is necessary for merging of files between releases. For example, if you are updating a system from FreeBSD 7.1 to FreeBSD 7.2, you will need to have update code built for FreeBSD 7.1-RELEASE and FreeBSD 7.2-RELEASE. Update client's *KeyPrint* and *Server Name* in `/etc/freebsd-update.conf`, and perform updates as instructed in the FreeBSD Update instructions in the handbook. The instructions can be found at *http://www.freebsd.org/doc/en/books/handbook/updating-freebsd-update.html*.

For reference, here is the entire run of `init.sh`.

Building a Patch

In the event a security advisory is posted to the FreeBSD Security Advisories page, *http://www.freebsd.org/security/advisories.html*, a patch update can be built. For this example, I will be using 7.1-RELEASE. A couple of assumptions are made for a different release build:

- Setup the correct directory structure for the initial build.
- Perform an initial build for 7.1-RELEASE.

Create patch directory under `/usr/local/freebsd-update-server/patches/` for the respective release.

Listing 5. Building Update Code

```
Mon Aug 24 17:54:07 PDT 2009 Extracting world+src for FreeBSD/amd64 7.2-RELEASE
Wed Sep 29 00:54:34 UTC 2010 Building world for FreeBSD/amd64 7.2-RELEASE
Wed Sep 29 01:49:42 UTC 2010 Distributing world for FreeBSD/amd64 7.2-RELEASE
Wed Sep 29 01:50:50 UTC 2010 Building and distributing kernels for FreeBSD/amd64 7.2-RELEASE
Wed Sep 29 02:02:56 UTC 2010 Constructing world components for FreeBSD/amd64 7.2-RELEASE
Wed Sep 29 02:03:08 UTC 2010 Distributing source for FreeBSD/amd64 7.2-RELEASE
Tue Sep 28 19:04:31 PDT 2010 Moving components into staging area for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 19:04:46 PDT 2009 Extracting extra docs for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 19:04:51 PDT 2009 Indexing world1 for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 19:08:04 PDT 2009 Locating build stamps for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 19:10:19 PDT 2009 Cleaning staging area for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 19:10:19 PDT 2009 Preparing to copy files into staging area for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 19:10:20 PDT 2009 Copying data files into staging area for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 12:16:57 PDT 2009 Copying metadata files into staging area for FreeBSD/amd64 7.2-RELEASE
Mon Aug 24 12:16:59 PDT 2009 Constructing metadata index and tag for FreeBSD/amd64 7.2-RELEASE

Files found which include build stamps:
kernel|generic|/GENERIC/hptrr.ko
kernel|generic|/GENERIC/kernel
world|base|/boot/loader
world|base|/boot/pxeboot
world|base|/etc/mail/freebsd.cf
world|base|/etc/mail/freebsd.submit.cf
world|base|/etc/mail/sendmail.cf
world|base|/etc/mail/submit.cf
world|base|/lib/libcrypto.so.5
world|base|/usr/bin/ntpq
world|base|/usr/include/osreldate.h
world|base|/usr/lib/libalias.a
world|base|/usr/lib/libalias_cuseeme.a
world|base|/usr/lib/libalias_dummy.a
world|base|/usr/lib/libalias_ftp.a
...
...
```

```
% mkdir -p /usr/local/freebsd-update-server/patches/RELEASE-7.1/
```

As an example, take the patch for named(8) found at *http://www.freebsd.org/cgi/man.cgi?query=named&sektion=8*. Read the advisory, and grab the necessary file from FreeBSD Security Advisories at *http://www.freebsd.org/security/advisories.html*. If you have trouble interpreting the advisory, please read this page for more information: *http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/security-advisories.html*.

Listing 6. Building Update Code

```
Values of build stamps, excluding library archive headers:
v1.2 (Aug 25 2009 00:40:36)
v1.2 (Aug 25 2009 00:38:22)
@(#)FreeBSD 7.2-RELEASE #0: Tue Aug 25 00:38:29 UTC 2009
FreeBSD 7.2-RELEASE #0: Tue Aug 25 00:38:29 UTC 2009
    root@server.myhost.com:/usr/obj/usr/src/sys/GENERIC
7.2-RELEASE
Mon Aug 24 23:55:25 UTC 2009
Mon Aug 24 23:55:25 UTC 2009
##### built by root@server.myhost.com on Tue Aug 25 00:16:15 UTC 2009
##### built by root@server.myhost.com on Tue Aug 25 00:16:15 UTC 2009
##### built by root@server.myhost.com on Tue Aug 25 00:16:15 UTC 2009
##### built by root@server.myhost.com on Tue Aug 25 00:16:15 UTC 2009
Mon Aug 24 23:46:47 UTC 2009
ntpq 4.2.4p5-a Mon Aug 24 23:55:53 UTC 2009 (1)
 * Copyright (c) 1992-2009 The FreeBSD Project.
Mon Aug 24 23:46:47 UTC 2009
Mon Aug 24 23:55:40 UTC 2009
Aug 25 2009
ntpd 4.2.4p5-a Mon Aug 24 23:55:52 UTC 2009 (1)
ntpdate 4.2.4p5-a Mon Aug 24 23:55:53 UTC 2009 (1)
ntpdc 4.2.4p5-a Mon Aug 24 23:55:53 UTC 2009 (1)
Tue Aug 25 00:21:21 UTC 2009
Tue Aug 25 00:21:21 UTC 2009
Tue Aug 25 00:21:21 UTC 2009
Mon Aug 24 23:46:47 UTC 2009
FreeBSD/amd64 7.2-RELEASE initialization build complete.  Please
review the list of build stamps printed above to confirm that
they look sensible, then run
# sh -e approve.sh amd64 7.2-RELEASE
to sign the release.
```

Listing 7. Mounting the key with the passphrase supplied

```
 # sh -e scripts/approve.sh amd64 7.2-RELEASE
 Wed Aug 26 12:50:06 PDT 2009 Signing build for FreeBSD/amd64 7.2-RELEASE
 Wed Aug 26 12:50:06 PDT 2009 Copying files to patch source directories for
FreeBSD/amd64 7.2-RELEASE
 Wed Aug 26 12:50:06 PDT 2009 Copying files to upload staging area for
FreeBSD/amd64 7.2-RELEASE
 Wed Aug 26 12:50:07 PDT 2009 Updating databases for FreeBSD/amd64 7.2-
RELEASE
 Wed Aug 26 12:50:07 PDT 2009 Cleaning staging area for FreeBSD/amd64 7.2-
RELEASE
```

From the security brief found here http://security.freebsd.org/advisories/FreeBSD-A-09:12.bind.asc, we can see it is called SA-09:12.bind.

After downloading the file, it is required to rename the file to an appropriate patch level. It is suggested to keep this inline with official FreeBSD patch levels, however, this is just a suggestion. For this build, let us follow the brief and call this p7. Rename the file:

```
% cd /usr/local/freebsd-update-server/patches/RELEASE-7.1/; mv bind.patch 7-SA-09:12.bind
```

Note

When running a patch level build, we are assuming that previous patches are in place. When a patch build is run, it will run all patches less than or equal to the number specified. Beyond this, you will have to take appropriate measures to verify authenticity of the patch.

You can also add your own patches to any build. Use the number zero, or any other number.

At this point, a diff is ready to be built. The software checks first to see if a `scripts/init.sh` has been run on the respective release prior to running the diff build.

```
# cd /usr/local/freebsd-update-server
# sh scripts/diff.sh amd64 RELEASE-7.1 7
```

What follows is the results of a diff build run (see Listing 8).

Note

Updates are printed, and approval is requested.

Follow the same process as noted before for appoving a build (see Listing 10).

After approving the build, upload the software.

```
# cd /usr/local/freebsd-update-server
# sh scripts/upload.sh amd64 RELEASE-7.1
```

For reference, here is the entire run of `diff.sh`.

Listing 8. Results of a diff buildrun

```
 # sh -e scripts/diff.sh amd64 7.1-RELEASE 7
 Wed Aug 26 10:09:59 PDT 2009 Extracting world+src for
FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 17:10:25 UTC 2009 Building world for
FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 18:05:11 UTC 2009 Distributing world for
FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 18:06:16 UTC 2009 Building and distributing
kernels for FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 18:17:50 UTC 2009 Constructing world
components for FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 18:18:02 UTC 2009 Distributing source for
FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 11:19:23 PDT 2009 Moving components into
staging area for FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 11:19:37 PDT 2009 Extracting extra docs for
FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 11:19:42 PDT 2009 Indexing world0 for
FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 11:23:02 PDT 2009 Extracting world+src for
FreeBSD/amd64 7.1-RELEASE-p7
 Thu Sep 30 18:23:29 UTC 2010 Building world for
FreeBSD/amd64 7.1-RELEASE-p7
 Thu Sep 30 19:18:15 UTC 2010 Distributing world for
FreeBSD/amd64 7.1-RELEASE-p7
 Thu Sep 30 19:19:18 UTC 2010 Building and distributing
kernels for FreeBSD/amd64 7.1-RELEASE-p7
 Thu Sep 30 19:30:52 UTC 2010 Constructing world
components for FreeBSD/amd64 7.1-RELEASE-p7
 Thu Sep 30 19:31:03 UTC 2010 Distributing source for
FreeBSD/amd64 7.1-RELEASE-p7
 Thu Sep 30 12:32:25 PDT 2010 Moving components into
staging area for FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 12:32:39 PDT 2009 Extracting extra docs for
FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 12:32:43 PDT 2009 Indexing world1 for
FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 12:35:54 PDT 2009 Locating build stamps for
FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 12:36:58 PDT 2009 Reverting changes due to
build stamps for FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 12:37:14 PDT 2009 Cleaning staging area for
FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 12:37:14 PDT 2009 Preparing to copy files
into staging area for FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 12:37:15 PDT 2009 Copying data files into
staging area for FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 12:43:23 PDT 2009 Copying metadata files into
staging area for FreeBSD/amd64 7.1-RELEASE-p7
 Wed Aug 26 12:43:25 PDT 2009 Constructing metadata
index and tag for FreeBSD/amd64 7.1-RELEASE-p7
 ...
```

```
 ...
 Files found which include build stamps:
 kernel|generic|/GENERIC/hptrr.ko
 kernel|generic|/GENERIC/kernel
 world|base|/boot/loader
 world|base|/boot/pxeboot
 world|base|/etc/mail/freebsd.cf
 world|base|/etc/mail/freebsd.submit.cf
 world|base|/etc/mail/sendmail.cf
 world|base|/etc/mail/submit.cf
 world|base|/lib/libcrypto.so.5
 world|base|/usr/bin/ntpq
 world|base|/usr/include/osreldate.h
 world|base|/usr/lib/libalias.a
 world|base|/usr/lib/libalias_cuseeme.a
 world|base|/usr/lib/libalias_dummy.a
 world|base|/usr/lib/libalias_ftp.a
 ...
 ...
 Values of build stamps, excluding library archive headers:
 v1.2 (Aug 26 2009 18:13:46)
 v1.2 (Aug 26 2009 18:11:44)
 @(#)FreeBSD 7.1-RELEASE-p7 #0: Wed Aug 26 18:11:50 UTC
2009
 FreeBSD 7.1-RELEASE-p7 #0: Wed Aug 26 18:11:50 UTC 2009
     root@server.myhost.com:/usr/obj/usr/src/sys/GENERIC
 7.1-RELEASE-p7
 Wed Aug 26 17:29:15 UTC 2009
 Wed Aug 26 17:29:15 UTC 2009
 ##### built by root@server.myhost.com on Wed Aug 26 17:
49:58 UTC 2009
 ##### built by root@server.myhost.com on Wed Aug 26 17:
49:58 UTC 2009
 ##### built by root@server.myhost.com on Wed Aug 26 17:
49:58 UTC 2009
 ##### built by root@server.myhost.com on Wed Aug 26 17:
49:58 UTC 2009
 Wed Aug 26 17:20:39 UTC 2009
 ntpq 4.2.4p5-a Wed Aug 26 17:29:42 UTC 2009 (1)
  * Copyright (c) 1992-2009 The FreeBSD Project.
 Wed Aug 26 17:20:39 UTC 2009
 Wed Aug 26 17:29:30 UTC 2009
 Aug 26 2009
 ntpd 4.2.4p5-a Wed Aug 26 17:29:41 UTC 2009 (1)
 ntpdate 4.2.4p5-a Wed Aug 26 17:29:42 UTC 2009 (1)
 ntpdc 4.2.4p5-a Wed Aug 26 17:29:42 UTC 2009 (1)
 Wed Aug 26 17:55:02 UTC 2009
 Wed Aug 26 17:55:02 UTC 2009
 Wed Aug 26 17:55:02 UTC 2009
 Wed Aug 26 17:20:39 UTC 2009
 ...
 ...
```

Listing 9. Building a patch

```
New updates:
kernel|generic|/GENERIC/kernel.symbols|f|0|0|0555|0|7c8dc176763f96ced0a57fc04e7c1b8d793f27e006dd13e0b499e1474ac47e10|
kernel|generic|/GENERIC/kernel|f|0|0|0555|0|33197e8cf15bbbac263d17f39c153c9d489348c2c534f7ca1120a1183dec67b1|
kernel|generic|/|d|0|0|0755|0||
src|base|/|d|0|0|0755|0||
src|bin|/|d|0|0|0755|0||
src|cddl|/|d|0|0|0755|0||
src|contrib|/contrib/bind9/bin/named/update.c|f|0|10000|0644|0|4d434abf0983df9bc47435670d307fa882ef4b348ed8ca90928d25
0f42ea0757|
src|contrib|/contrib/bind9/lib/dns/openssldsa_link.c|f|0|10000|0644|0|c6805c39f3da2a06dd3f163f26c314a4692d4cd9a2d929c
0acc88d736324f550|
src|contrib|/contrib/bind9/lib/dns/opensslrsa_link.c|f|0|10000|0644|0|fa0f7417ee9da42cc8d0fd96ad24e7a34125e05b5ae075b
d6e3238f1c022a712|
...
...
FreeBSD/amd64 7.1-RELEASE update build complete.  Please review
the list of build stamps printed above and the list of updated
files to confirm that they look sensible, then run
# sh -e approve.sh amd64 7.1-RELEASE
to sign the build.
```

Listing 10. Approving a build

```
# sh -e scripts/approve.sh amd64 7.1-RELEASE
Wed Aug 26 12:50:06 PDT 2009 Signing build for FreeBSD/amd64 7.1-RELEASE
Wed Aug 26 12:50:06 PDT 2009 Copying files to patch source directories for FreeBSD/amd64 7.1-RELEASE
Wed Aug 26 12:50:06 PDT 2009 Copying files to upload staging area for FreeBSD/amd64 7.1-RELEASE
Wed Aug 26 12:50:07 PDT 2009 Updating databases for FreeBSD/amd64 7.1-RELEASE
Wed Aug 26 12:50:07 PDT 2009 Cleaning staging area for FreeBSD/amd64 7.1-RELEASE
The FreeBSD/amd64 7.1-RELEASE update build has been signed and is
ready to be uploaded.  Remember to run
# sh -e umountkey.sh
to unmount the decrypted key once you have finished signing all
the new builds.
```

Tips

- If you build your own release using the native make release, freebsd-update-server code will work from your release. As an example, you may build a release without ports or documentation and add a custom kernel. After removing functionality pertaining to the documentation subroutine and altering the `buildworld()` subroutine in scripts/build.subr the freebsd-update-code will successfully build update code on this release.
- Add make `-j NUMBER` to `scripts/build.subr` to speed up processing. Adding flags to anything other than make buildworld and make obj may cause the build to become unreliable.
- Create a firewall rule to block outgoing RST packets. Due to a bug noted in this posting, *http://unix.derkeiler.com/Mailing-Lists/FreeBSD/stable/2009-04/msg00365.html*, you will have many time-outs and fail to update a system.
- Create an appropriate DNS SRV record for your update server, and put others behind it with variable weights. This effectively creates update mirrors.

```
_http._tcp.update.myserver.com.   IN
SRV    0 2 80    host1.myserver.com.
SRV    0 1 80    host2.myserver.com.
SRV    0 0 80    host3.myserver.com.
```

- Please read the source documentation. It contains valuable information that will allow you to utilize all features of the software.

Afterword

This FreeBSD Update article, found at *http://www.experts-exchange.com/articles/OS/Unix/BSD/FreeBSD/Build-Your-Own-FreeBSD-Update-Server.html*, was originally published at Experts-Exchange (*http://www.experts-exchange.com*).

WEBHOSTINGBUZZ.COM

800.252.1887

sales@webhostingbuzz.com

Visit www.webhostingbuzz.com
to see all plans or order

Shared Hosting

FreeBSD powered shared web hosting with a free domain name and cPanel control panel. Perfect for smaller websites and prices from $4.95 per month

750GB disk space
15000GB bandwidth
1 Free Domain Name
24 x 7 phone, chat and helpdesk support

from $**7.95** per month

Virtual Private Servers

Full Control: Root access, power on/off/reboot at any time.
Choice of operating systems including FreeBSD
Dedicated Resources: Guaranteed share of CPU and RAM
Scalability: Buy only the resources you need now, and upgrade them as demand increases with virtually limitless capacity

CPU 1Ghz Guaranteed
512MB RAM Guaranteed
50GB Disk Space
FreeBSD

$**34.95** per month

Dedicated Servers

Our completely customizable range of Managed Services allows you to focus on running your business, while knowing a team of IT professionals are hard at work managing your IT environment.

- Dedicated servers to fit any budget
- Fully managed clusters
- High availability

Dual Xeon 5520
8GB RAM
2 x 1.5TB HDD
FreeBSD

$**549** per month

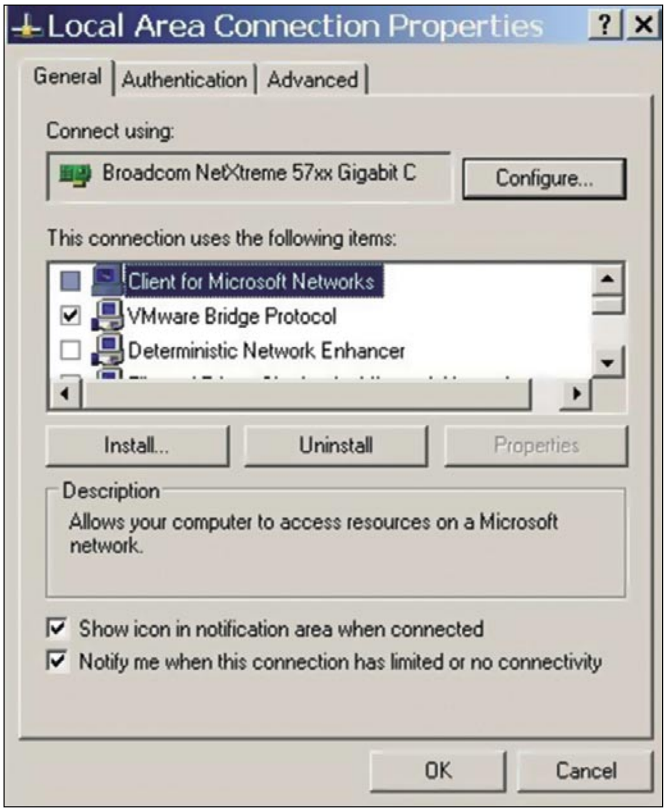
Using OpenBSD and PF as a Virtual Firewall for Windows

Pedro Lereno

This article describes how to protect a Windows host with a basic configuration of an OpenBSD virtual machine with PF as a NAT router and firewall.

With the increasing usage of third-party networks (like hotel networks and wireless hotspots) people are increasingly putting their Windows laptops at risk. When we connect to unknown networks, we lose the protection of our home NAT router or enterprise firewall.

The Windows firewall, by default, has many open ports to the local network, like the file and print sharing service ports, which are the source of many security holes. Some malware changes settings on the Windows firewall and hides those changes. There is nothing better than a different OS in the middle to keep track of our network traffic.



Local Area Connection Properties

General | Authentication | Advanced

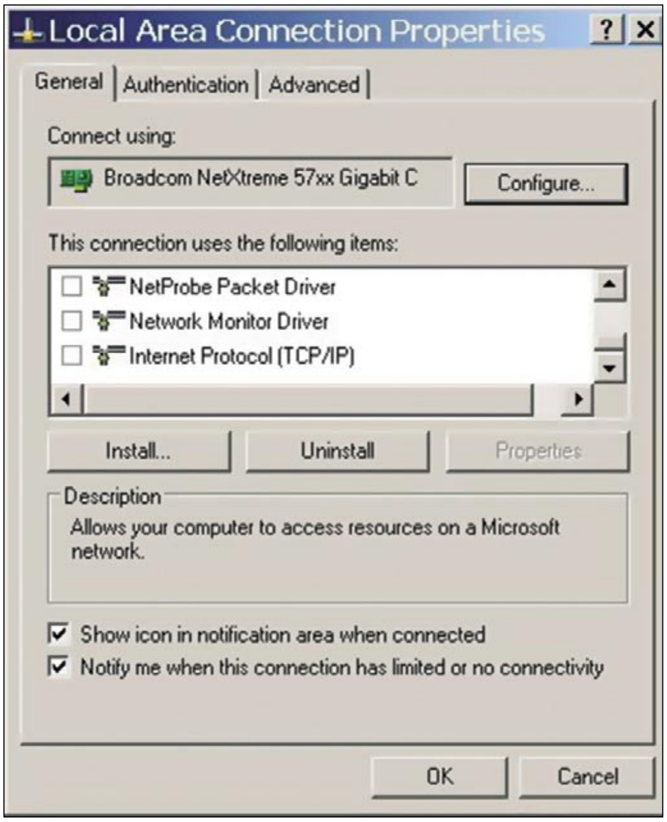
Connect using: Broadcom NetXtreme 57xx Gigabit C

This connection uses the following items:
- [ ] Client for Microsoft Networks
- [✓] VMware Bridge Protocol
- [ ] Deterministic Network Enhancer

Description: Allows your computer to access resources on a Microsoft network.

- [✓] Show icon in notification area when connected
- [✓] Notify me when this connection has limited or no connectivity

Figure 1. Unselect all items except VMware Bridge Protocol



Local Area Connection Properties

General | Authentication | Advanced

Connect using: Broadcom NetXtreme 57xx Gigabit C

This connection uses the following items:
- [ ] NetProbe Packet Driver
- [ ] Network Monitor Driver
- [ ] Internet Protocol (TCP/IP)

Description: Allows your computer to access resources on a Microsoft network.

- [✓] Show icon in notification area when connected
- [✓] Notify me when this connection has limited or no connectivity

Figure 2. Be sure to unselect Windows TCP/IP

Because of that a new kind of device has been born: portable travel routers. With that in mind, why not create our own *travel router* inside our laptop.

After reading the great article from Prof. Vassilis Prevalakis from Drexel University in Philadelphia (*http://www.prevelakis.net/Papers/VirtualFirewall.pdf*), I decided to build my own virtual firewall.

Preparing the Host Machine

First we need to install a new VMware Server virtual machine. After the machine is built and configured with the correct hardware interfaces we can deploy to other hosts with the lightweight VMware Player.

I have chosen VMware because of my experience with it. I didn't try this setup with other virtualization products but I think it will work in the same way.

To get VMware go to *http://www.vmware.com*, select Products and VMware Server (a free virtual server). To download the product you have to register. Once downloaded, install VMware.

In order to make your Windows host invisible to the outside world, configure your Windows network adapter with only the *VMware Bridge Protocol* selected (Figure 1). Be sure to unselect *Client for Microsoft Networks* and *Internet Protocol (TCP/IP)* (see Figure 2).

On Vmnet1, configure the default gateway and the DNS servers. The default gateway will be the IP address of the *host only* interface (vic1) of the *virtual machine* (VM). The other interface of the VM is *bridged* with the real interface (Figure 3). The Vmnet8 Windows interface may be disabled.

The *host only* interface connects only to the host; the *bridged* interface is shared with the physical interface.

We can check the Windows IP configuration with the command ipconfig:

Ethernet adapter VMware Network Adapter VMnet1:

```
        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . .
: 192.168.21.1
        Subnet Mask . . . . . . . . . . .
: 255.255.255.0
        Default Gateway . . . . . . . . .
: 192.168.21.2
```

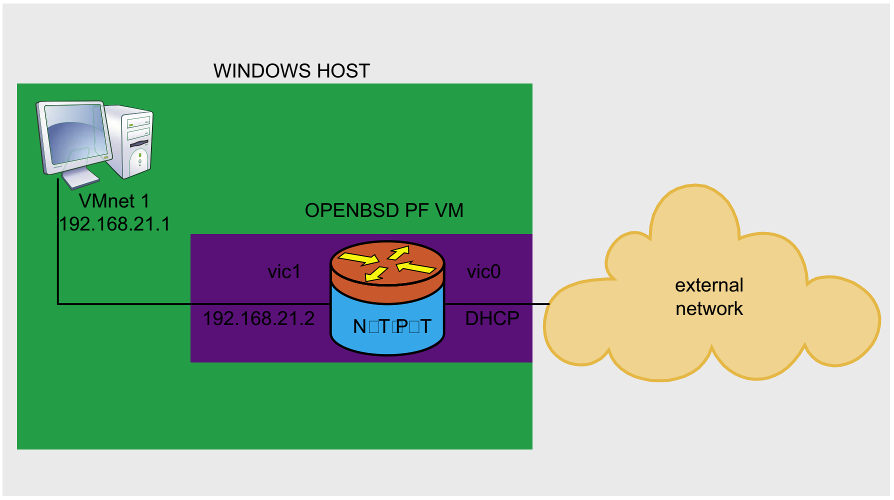
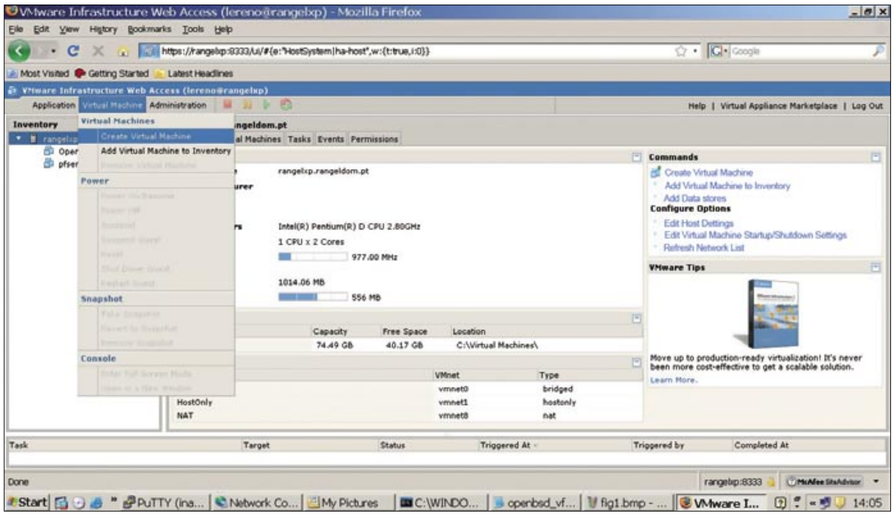


Figure 3. Virtual scenario: Windows host connected to NAT router


VMware Infrastructure Web Access (lereno@rangelxp) - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Application Virtual Machine Administration
Help | Virtual Appliance Marketplace | Log Out
Inventory
Virtual Machines
Create Virtual Machine
Add Virtual Machine to Inventory
Power
Snapshot
Console
Tasks Events Permissions
Commands
Create Virtual Machine
Add Virtual Machine to Inventory
Add Data stores
Configure Options
Edit Host Dettings
Edit Virtual Machine Startup/Shutdown Settings
Refresh Network List
VMware Tips
Learn More.
Start PuTTY (ina... Network Co... My Pictures C:\WINDO... openbsd_vf... fig1.bmp - ... VMware I... 14:05

https://rangelxp:8333/ui/#{e:"HostSystem|ha-host",w:{t:true,i:0}}

rangelxp.rangeldom.pt

Intel(R) Pentium(R) D CPU 2.80GHz
1 CPU x 2 Cores
977.00 MHz
1014.06 MB
556 MB

Capacity	Free Space	Location
74.49 GB	40.17 GB	C:\Virtual Machines\

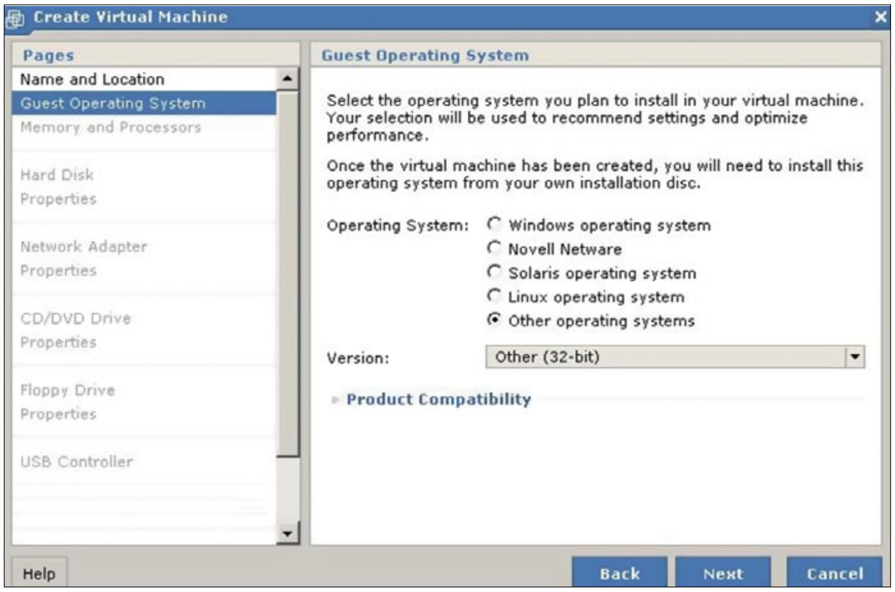
	VMnet	Type
	vmnet0	bridged
HostOnly	vmnet1	hostonly
NAT	vmnet8	nat

Move up to production-ready virtualization! It's never been more cost-effective to get a scalable solution.

Task	Target	Status	Triggered At	Triggered by	Completed At

Done

Figure 4. VMware Server: Virtual Machine creation



Guest Operating System

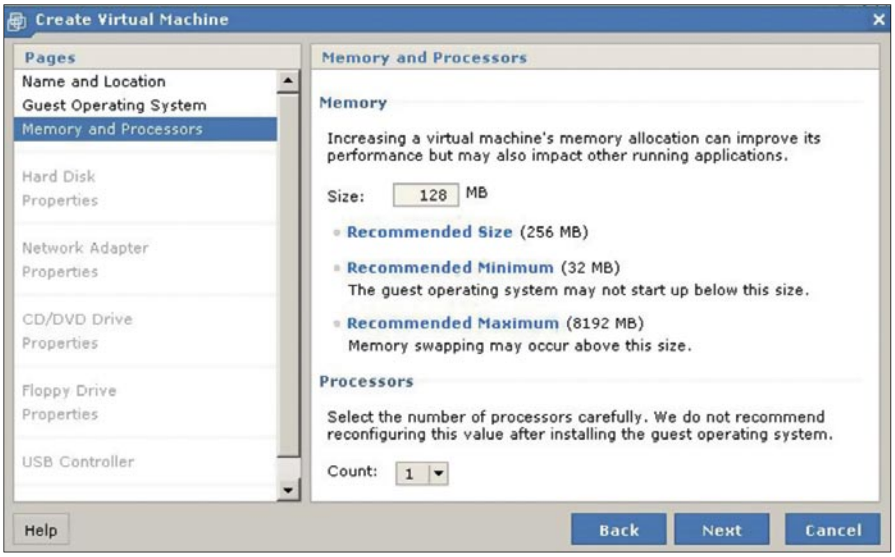
Select the operating system you plan to install in your virtual machine. Your selection will be used to recommend settings and optimize performance.

Once the virtual machine has been created, you will need to install this operating system from your own installation disc.

Operating System: Windows operating system / Novell Netware / Solaris operating system / Linux operating system / [✓ Other operating systems]

Version: Other (32-bit)

Figure 5. VMware Server: Guest Operating System selection



Memory and Processors

Memory

Increasing a virtual machine's memory allocation can improve its performance but may also impact other running applications.

Size: 128 MB

Recommended Size (256 MB)

Recommended Minimum (32 MB)
The guest operating system may not start up below this size.

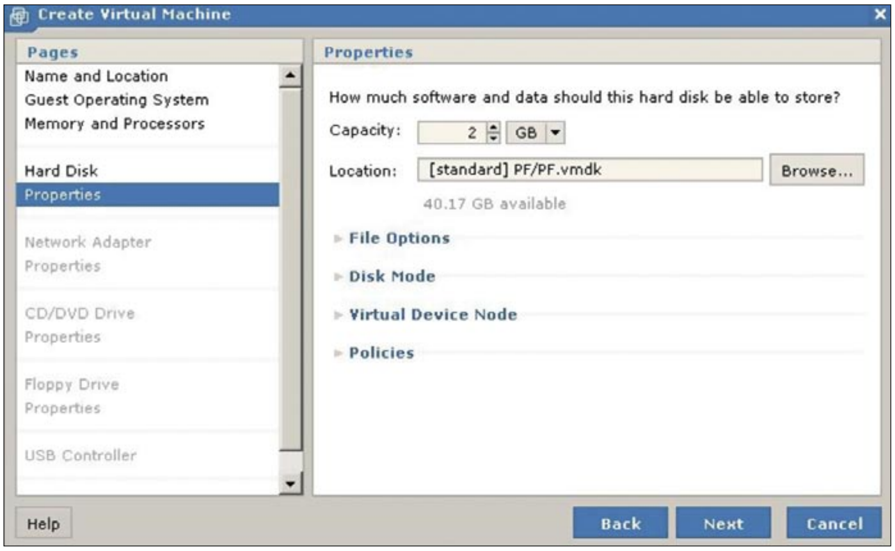
Recommended Maximum (8192 MB)
Memory swapping may occur above this size.

Processors

Select the number of processors carefully. We do not recommend reconfiguring this value after installing the guest operating system.

Count: 1

Figure 6. VMware Server: Virtual Machine memory and processor



Properties

How much software and data should this hard disk be able to store?

Capacity: 2 GB

Location: [standard] PF/PF.vmdk

40.17 GB available

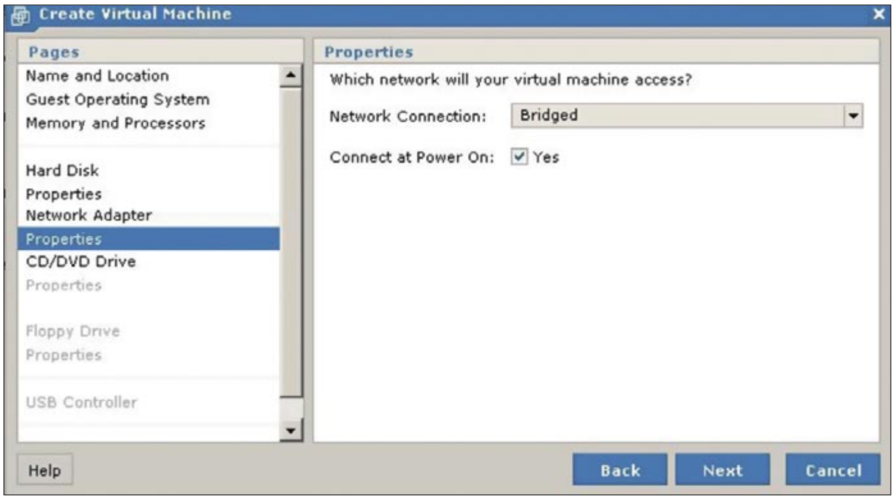
File Options

Disk Mode

Virtual Device Node

Policies

Figure 7. VMware Server: Virtual Machine disk configuration



Properties

Which network will your virtual machine access?

Network Connection: Bridged

Connect at Power On: ✓ Yes

Figure 8. VMware Server: Virtual Machine network connection

There is no IP configuration on the external interface.

Installing VMware Server

Open VMware Infrastructure Web access and authenticate with your Windows account. Go to Virtual Machine menu and select *Create Virtual machine* (see Figure 4).

Give a name to this virtual machine (in this example it was *PF*.)

Select the operating system: *Other operating System: Other (32-bit)* (see Figure 5).

Select memory size 128 MB, use of 1 processor (see Figure 6). Create a new virtual disk with 2 GB (see Figure 7), default settings. Add Network Adapter, one network connection *bridged* (Figure 8). Use the CD physical drive, or the ISO image to install the operating system (install45.iso from *http://openbsd.org*). Don't have floppy. The USB controller might be useful to direct connect USB network interfaces (for example 3G modems). After finishing this wizard, go to the virtual machine summary page and on the right side menu select *Add hardware*, select network adapter and then choose network connection HostOnly (see Figure 9).

This is the summary of the created virtual machine (see Figure 10). Power On.

Installing OpenBSD

I have chosen OpenBSD and PF for the virtual appliance because of the strong security of the OS and the easy to understand commands of PF (for those used to Cisco, these rules are familiar). On the listing 1 you can see the installation of OpenBSD. Pay attention to the interface cards IP addresses. The system has only one system partition and the swap for an easy install and understanding.

Installing Some Tools and PF Configuration

Uncomment the following line of the file of `/etc/sysctl.conf` by deleting the `#`:

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

This option permits the traffic flow between the two interfaces. We are not considering using multicast and ipv6, but if necessary, uncomment those lines too. Activate PF in `/etc/rc.conf.local`:

```
pf=YES
```

Restart to activate the changes. Add the following line to `/etc/hosts` file:

```
192.168.21.1    localpc
```

Install these packages that can be useful for monitoring network traffic from the internet.

```
# export PKG_PATH=ftp://
ftp3.usa.openbsd.org/pub/OpenBSD/4.5/
packages/i386
# pkg_add pftop
# pkg_add ntop
```

If you are familiar with *top* for process management, these tools work in the same format and can be useful to track network connections and firewall activity. This is an example of a very simple `pf.conf` configuration:

```
/etc/pf.conf
scrub in
nat on vic0 from vic1:network to any
-> vic0
block in all
pass in on vic1
```

Testing the firewall rules:

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

adding the rules:

```
#pfctl -vf /etc/pf.conf   # v-verbose
output
```

We are blocking all incoming traffic from the external interface, permitting traffic from the internal interface, and making NAT. Now we have access from our `internal` Windows host to the outside world. We can try *nmap* from the outside to check for open ports.

Connect to the Corporate VPN

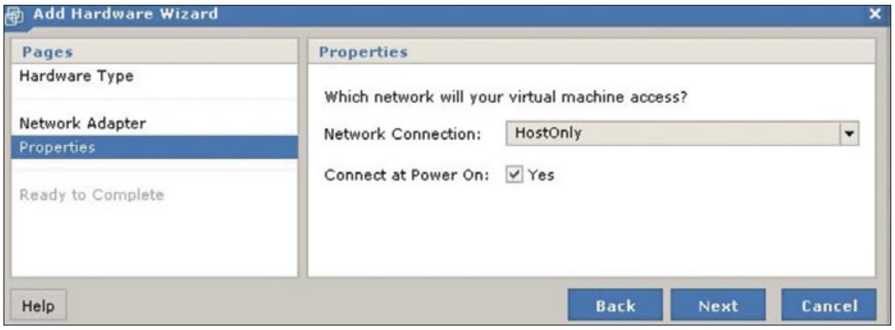
Our corporate firewall is a Cisco ASA. Our remote teleworkers connect through Cisco VPN client software. To resolve this problem while not opening unnecessary holes in our virtual firewall, we use the package vpnc. Installing the package:

```
#export PKG_PATH=ftp://ftp3.usa.open
bsd.org/pub/OpenBSD/4.5/packages/i386
#pkg_add vpnc
```

Use a new pf configuration (NAT has changed because of VPN):

Listing 1. OpenBSD installation on the Virtual Machine

```
choose install (i)
Terminal type vt220
keyboard mapping, in my case was pt
Procced with install? yes
Which one of the root disk? sd0
Use all disk for OpenBSD
> a b
offset: [63]
size: [4192902] 256m
Rounding to cylinder: 530082
FS type: [swap]
> a a
offset [530145]
size [3662820]
FS type: [4.2BSD]
mount point: [none] /
>q
Write new label?: y
Are you really sure that you are ready to proceed? yes
System hostname? PF
Configure network? yes
Available interfaces are: vic0 vic1.
Which one do you wish to initialize? vic0
Symbolic name for vic0? pf_ext
Do you want to change the media options? no
IPv4 address for vic0? dhcp
IPv6 for vic0? none
Available interfaces are: vic1.
Which one do you wish to initialize? vic1
Symbolic name for vic1? pf_int
Do you want to change the media options? no
IPv4 address for vic1? 192.168.21.2
Netmask? 255.255.255.0
IPv6 for vic1? none
DNS domain name? (put yor default domain)
DNS nameserver? 208.67.222.222 208.67.220.220 (example
from opendns)
Use the nameserver now? yes
Default IPv4 route? dhcp
Edit host with ed? no
Do you want to do any manual network configuration? no
Password for root account? ....
Location of the sets? cd
Which one contains the install media? cd0
Pathname to the sets? 4.5/i386
Set name? done (leave the default)
Ready to install sets? yes
....
Locations of the sets? done
Start ssh by default? yes
Start ntpd by default? no
Do you expect to run X Windows system? no
Change the default console to com0? no
What timezone are you in? Portugal (choose your own)
halt
unplug the iso cd (point to physical) and reset your
virtual machine
```


Add Hardware Wizard

Pages
- Hardware Type
- Network Adapter
- Properties
- Ready to Complete

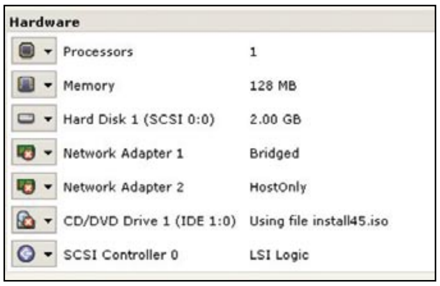
Properties

Which network will your virtual machine access?

Network Connection: HostOnly

Connect at Power On: ☑ Yes

Figure 9. VMware Server: adding hardware to the Virtual Machine


Hardware

Processors	1
Memory	128 MB
Hard Disk 1 (SCSI 0:0)	2.00 GB
Network Adapter 1	Bridged
Network Adapter 2	HostOnly
CD/DVD Drive 1 (IDE 1:0)	Using file install45.iso
SCSI Controller 0	LSI Logic

Figure 10. VMware Server: Virtual Machine hardware configuration

```
set skip on lo
set skip on tun0
scrub in
nat on tun0 from !(tun0) to any ->
(tun0)
block in log all
pass on tun0
pass from {lo0, vic1:network, vic0} to
any keep state
```

Fill the file /etc/vpnc/default.conf with your Cisco VPN client access profile. Start the vpn:

```
#vpnc
#ifconfig tun0 mtu 1452
```

The default mtu size for tun0 is 1414, this is sufficient if our host was the virtual appliance, but is insufficient for the NAT'ed host before (Figure 11).

You can see the encrypted ipsec network traffic with command:

```
#tcpdump -e -ttt -n -i vic0
```

Or the unencrypted traffic:

```
#tcpdump -e -ttt -n -i vic1
```

Conclusion

This method is not as safe as having an external device, but it is safer than many default router configurations with, for example, *Universal Plug-and-Play* (UPnP) enabled.

We can make it a lot safer from unexpected Windows behavior, by using a non-admin account without privileges to change network interfaces and VMware Authentication service to boot the VM as a non-Administrator account.

Those not familiar with command-line configuration of PF, can use Pfsense. Pfsense is a FreeBSD distribution customized for routers and firewalls with a nice web interface to manipulate PF rules. After booting the ISO live image available on VMware Server or using the VMware appliance in VMware Player, we choose le0 (bridged) to WAN interface and le1 (host-only) to LAN, then all the configuration is done on the browser.

All of these examples were made with Microsoft Windows XP Professional (US English) as the host system and OpenBSD 4.5 in the Virtual Machine. My objective was to describe the preparation of the host system, the VM, and the way they interconnect step by step. Many more things can be done with PF that I didn't mention. Read the references for going deeper into PF.

This can also be a good way to test complex firewall rules before applying them to the corporate firewall. Or, if you cannot get rid of your Windows desktop because of some applications, it is a good way to get the best of both worlds.

I hope you enjoy the Virtual Firewall.

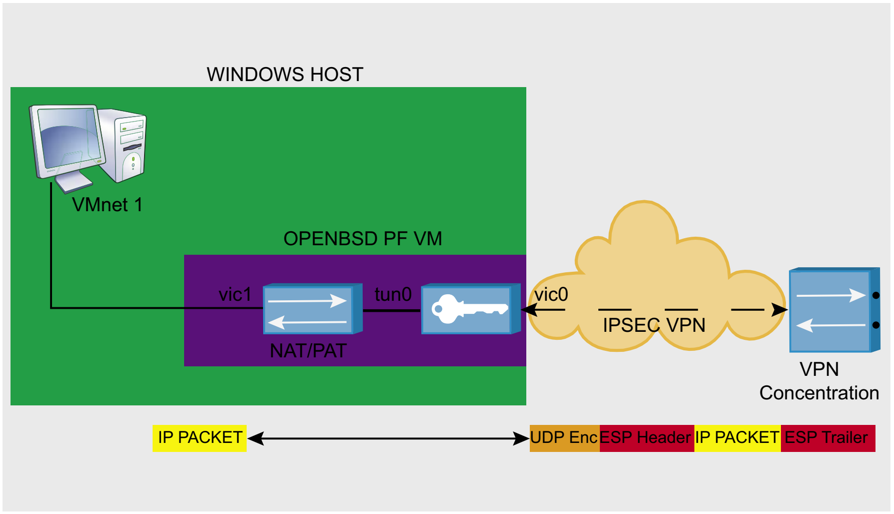


Figure 11. VPN encapsulation on the Virtual Router

References

- *The Virtual Firewall* – Vassilis Prevalakis – *www.prevelakis.net/Papers/VirtualFirewall.pdf*
- *Firewalling with OpenBSD's PF packet filter* – Peter Hansteen – *http://www.bsdly.net/~peter/pf.html*
- *PF User Guide – http://www.openbsd.org/faq/pf/index.html*
- *Pfsense – http://www.pfsense.com/*

Looking for help, tip or advice?
Want to share your knowledge with others?
Visit BSD magazine forum
BSD MAGAZINE
Give us your opinion about the magazine's content and help us create the most useful source for you!
www.bsdmag.org

Keeping FreeBSD Applications Up-To-Date

Richard Bejtlich
Principal Technologist and Director of Incident Response, General Electric

An important system administration task, and a principle of running a defensible network, is keeping operating systems and applications up-to-date.

Running current software is critical when older services are vulnerable to exploitation. Obtaining new features not found in older applications is another reason to run current software. Fortunately, open source software offers a variety of means to give users a secure, capable computing environment.

This article presents multiple ways to keep FreeBSD applications up-to-date. I use a FreeBSD 7.1 system, and subsequent versions, to demonstrate how to install applications not included with the OS and how to keep those applications up-to-date. It is important to realize that this article discusses applications only; it does not discuss the OS. FreeBSD does not have a unified update mechanism for the OS and applications. By applications I mean software outside of the kernel and userland. For example, Debian systems can use the apt tool to keep the distribution and packaged applications up-to-date. FreeBSD does not have a single equivalent tool, so this article only addresses keeping applications up-to-date.

In this article I do not differentiate between an update and an upgrade. I will use the term update to describe any action that changes the version of an installed application.

I chose FreeBSD 7.1, released in January 2009, as my starting point because applications for it offer a security history suitable for describing multiple update cases. At the time of writing FreeBSD 7.2 is the latest STABLE release and 8.0 is now available. Readers wondering why someone might want to install an *old* OS version can imagine that there might be an application supported only on FreeBSD 7.1 and not yet officially ready for 7.2 or 8.0, prompting an administrator to run a 7.1 box.

All of the work done in this article was done remotely via OpenSSH. One danger of performing remote upgrades is losing connection during a critical phase of the process. One software-based way to deal with this issue is to conduct all remote upgrades within a screen(1) session. (*http://www.freshports.org/misc/screen*) Should you lose connectivity during the upgrade while running screen, your session will continue uninterrupted. The screen(1) program has suffered security problems in the past, so balance its features against the possible risks.

My advice on administering this reference platform is based on deploying FreeBSD on servers, workstations, and laptops since 2000. The article represents a mix of my interpretations of official FreeBSD documentation, inputs from mentors, and the result of my own experimentation and deployment strategies. This guide cannot be anywhere near a complete reference on keeping FreeBSD up-to-date or maintaining a secure system. I strongly recommend reading the excellent FreeBSD Handbook as well as the multiple helpful published books on FreeBSD.

FreeBSD Handbook and Absolute FreeBSD 2nd Ed

Please note that Chapter 4, Installing Applications: Packages and Ports, is the authoritative source for information on keeping FreeBSD applications up-to-date (*http://www.freebsd.org/doc/en/books/handbook/ports.html*). The reason I wrote this article was to show how these various mechanisms apply in practice, and which I prefer in production. I must also recommend Michael W. Lucas' excellent book Absolute FreeBSD, 2nd Ed (No Starch, 2008). Several other excellent FreeBSD writers have produced books, but Michael's is my favorite. For deeper coverage on the topics in this article, please see the Handbook or Michael's book.

A Common Linux Experience

FreeBSD's application installation, maintenance, and removal process is sometimes confusing to those with a Linux background. For purposes of a brief comparison, I will demonstrate how to install the Curl application on a Debian 5.0 host using the apt-get tool. For authoritative documentation on using APT, please see *http://www.debian.org/doc/manuals/apt-howto/*. To install Curl, the user simply enters `apt-get install curl` (see Listing 1).

That is easy enough!

Simple Package Installation on FreeBSD

FreeBSD users can install Curl using a similar method (see Listing 2).

First we set a proxy for our environment. The `-v` switch permits seeing verbose output. The command to install the Curl package on FreeBSD from a remote package repository requires the `-r` switch. You can see the location from where the package was retrieved in this output:

```
document: [/pub/FreeBSD/ports/i386/
packages-7.1-release/Latest/curl.tbz]
...edited...
Fetching ftp://ftp.freebsd.org/pub/
FreeBSD/ports/i386/packages-7.1-
release/Latest/curl.tbz...
```

If you visit the FTP server and look at the directory, you'll see that `curl.tbz` is really a symlink to the following:

```
ftp://ftp.freebsd.org//pub/FreeBSD/
ports/i386/packages-7.1-release/All/
curl-7.18.0.tbz
```

The `packages-7.1-release` directory means that the package `curl-7.18.0.tbz` is the version of the package built for the release of FreeBSD 7.1, as was *shipped on CD*. Newer versions are available remotely and I will describe how to acquire those later.

The `pkg_info` command shows the Curl package is now installed. I issue the `rehash` command to ensure that curl is in the path for the user's shell.

Checking for Vulnerable Packages with Portaudit

FreeBSD's Portaudit tool is the easiest way to determine if any installed packages have security vulnerabilities. Portaudit relies on the FreeBSD VuXML site (*http://www.vuxml.org/freebsd/*) for knowledge of vulnerable packages. Don't worry about the term *port* vs. package right now; I'll address it soon. To see if the installed packages have any vulnerabilities, install and run Portaudit (see Listing 3). We see that Curl has a security vulnerability that requires a patch. We'll address ways to fix that in the following sections.

Listing 1. Installing curl on Debian using apt-get

```
shuttle02:~# uname -a

Linux shuttle02 2.6.26-1-686 #1 SMP Fri Mar 13 18:08:45 UTC 2009 i686 GNU/
Linux

shuttle02:~# apt-get install curl
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  ca-certificates libcurl3 libssh2-1 openssl
The following NEW packages will be installed:
  ca-certificates curl libcurl3 libssh2-1 openssl
0 upgraded, 5 newly installed, 0 to remove and 1 not upgraded.
Need to get 1687kB of archives.
After this operation, 4133kB of additional disk space will be used.
Do you want to continue [Y/n]? y
Get:1 http://http.us.debian.org stable/main openssl 0.9.8g-15+lenny1 [1036kB]
Get:2 http://security.debian.org stable/updates/main libcurl3 7.18.2-8lenny3
[228kB]
Get:3 http://http.us.debian.org stable/main ca-certificates 20080809 [151kB]
Get:4 http://http.us.debian.org stable/main libssh2-1 0.18-1 [64.3kB]
Get:5 http://security.debian.org stable/updates/main curl 7.18.2-8lenny3
[208kB]
Fetched 1687kB in 1s (1290kB/s)
Preconfiguring packages ...
Selecting previously deselected package openssl.
(Reading database ... 51192 files and directories currently installed.)
Unpacking openssl (from .../openssl_0.9.8g-15+lenny1_i386.deb) ...
Selecting previously deselected package ca-certificates.
Unpacking ca-certificates (from .../ca-certificates_20080809_all.deb) ...
Selecting previously deselected package libssh2-1.
Unpacking libssh2-1 (from .../libssh2-1_0.18-1_i386.deb) ...
Selecting previously deselected package libcurl3.
Unpacking libcurl3 (from .../libcurl3_7.18.2-8lenny3_i386.deb) ...
Selecting previously deselected package curl.
Unpacking curl (from .../curl_7.18.2-8lenny3_i386.deb) ...
Processing triggers for man-db ...
Setting up openssl (0.9.8g-15+lenny1) ...
Setting up ca-certificates (20080809) ...
Updating certificates in /etc/ssl/certs....done.
Running hooks in /etc/ca-certificates/update.d....done.
Setting up libssh2-1 (0.18-1) ...
Setting up libcurl3 (7.18.2-8lenny3) ...
Setting up curl (7.18.2-8lenny3) ...
shuttle02:~# which curl
/usr/bin/curl
```

Listing 2. Installing curl on FreeBSD using pkg_add

```
freebsd7# uname -a
FreeBSD freebsd7.localdomain 7.1-RELEASE FreeBSD 7.1-
RELEASE #0: Thu Aug 20 11:24:04 EDT 2009     root@freebs
d7.localdomain:/usr/obj/usr/src/sys/FREEBSD7  i386

freebsd7# setenv HTTP_PROXY http://172.16.2.1:3128
freebsd7# pkg_add -vr curl
scheme:   [ftp]
user:     []
password: []
host:     [ftp.freebsd.org]
port:     [0]
document: [/pub/FreeBSD/ports/i386/packages-7.1-release/
Latest/curl.tbz]
scheme:   [http]
user:     []
password: []
host:     [172.16.2.1]
port:     [3128]
document: [/]
---> 172.16.2.1:3128
looking up 172.16.2.1
connecting to 172.16.2.1:3128
requesting ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/
packages-7.1-release/Latest/curl.tbz
>>> GET ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/
packages-7.1-release/Latest/curl.tbz HTTP/1.1
>>> Host: ftp.freebsd.org
>>> User-Agent: pkg_add libfetch/2.0
>>> Connection: close
>>>
<<< HTTP/1.0 200 Gatewaying
<<< Server: squid/2.7.STABLE6
<<< Date: Mon, 24 Aug 2009 19:52:19 GMT
<<< Content-Type: text/plain
<<< Content-Length: 1088297
content length: [1088297]
<<< Last-Modified: Mon, 08 Sep 2008 10:45:09 GMT
last modified: [2008-09-08 10:45:09]
<<< X-Cache: MISS from r200a.taosecurity.com
<<< Via: 1.0 r200a.taosecurity.com:3128 (squid/
2.7.STABLE6)
<<< Connection: close
<<<
offset 0, length -1, size -1, clength 1088297
Fetching ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/
packages-7.1-release/Latest/curl.tbz...
x +CONTENTS
x +COMMENT
x +DESC
x +MTREE_DIRS
x man/man1/curl.1.gz
...edited...
x share/examples/curl/synctime.c
tar command returns 0 status
 Done.
extract: Package name is curl-7.18.0
extract: CWD to /usr/local
extract: /usr/local/man/man1/curl.1.gz
...edited...
extract: /usr/local/share/examples/curl/synctime.c
extract: execute '/sbin/ldconfig -m /usr/local/lib'
extract: CWD to .
Running mtree for curl-7.18.0..
mtree -U -f +MTREE_DIRS -d -e -p /usr/local >/dev/null
Attempting to record package into /var/db/pkg/curl-
7.18.0..
Package curl-7.18.0 registered in /var/db/pkg/curl-
7.18.0
freebsd7# pkg_info
curl-7.18.0         Non-interactive tool to get files
from FTP, GOPHER, HTTP(S)
freebsd7# rehash
```

Listing 3. Installing portaudit on FreeBSD

```
freebsd7# pkg_add -r portaudit
Fetching ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/
packages-7.1-release/Latest/portaudit.tbz... Done.
===>  To check your installed ports for known
vulnerabilities now, do:
      /usr/local/sbin/portaudit -Fda

freebsd7# rehash

freebsd7# portaudit -Fdav

Attempting to fetch from http://www.FreeBSD.org/ports/.
auditfile.tbz                                 100% of
57 kB   95 kBps
New database installed.
Database created: Mon Aug 24 15:10:03 EDT 2009
Affected package: curl-7.18.0 (matched by
curl>=5.11<7.19.4)
Type of problem: curl -- cURL/libcURL Location: Redirect
URLs Security Bypass.
Reference: <http://www.FreeBSD.org/ports/portaudit/
5d433534-f41c-402e-ade5-e0a2259a7cb6.html>

1 problem(s) in your installed packages found.

You are advised to update or deinstall the affected
package(s) immediately.
```

FreeBSD Package Repositories

It is important to understand what version of packages are made available through the FreeBSD project. Visiting *ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/* shows what's available for the i386 platform. (The FreeBSD team also regularly builds packages for the amd64 platform, see Listing 4)

For the purposes of our system (running a version of FreeBSD 7.x), we care about the packages-7* directories.

Earlier we installed Curl and it was retrieved from the packages-7.1-release directory. The packages-7.2-release directory is likely to contain a newer version of Curl since 7.2 was released months after 7.1. If we check that directory, we find `curl-7.19.4.tbz` is available, with a build date of Apr 21.

Looking back at our Portaudit output, we see that the 7.19.4 is not vulnerable

```
(matched by curl>=5.11<7.19.4)
```

Let's remember that for now, but also look at the other packages-7* directory, packages-7-stable. In that directory we find `curl-7.19.6_1.tbz` available, with a build date of Aug 22. That version is also not vulnerable.

So what does packages-7-stable mean? That directory contains the latest packages built for FreeBSD 7.x. If you're thinking that you might want to install packages from that site on a regular basis, you are right. I'll cover that soon. For now we want to know how to update Curl to a newer version.

Updating Packages by Deletion and Addition

Deleting an installed package and adding a new version is one way to update a package. The easiest way to accomplish this goal is to change to the `/var/db/pkg` directory and use the `pkg_delete` command (see Listing 5).

With Curl deleted, we can add the new version. For demonstration purposes we'll add the version shipped with FreeBSD 7.2 RELEASE. To tell `pkg_add` how to get that package, we set the PACKAGESITE variable (see Listing 6).

Curl is now installed. Notice that a dependency, `ca_root_nss`, was also installed. If we rerun Portaudit, the vulnerability should be eliminated (see Listing 7).

That process seems simple enough. However, it is probably not convenient to delete and add every package on a system when the administrator wants to update the packages. To run a more automated update system, we have to turn to the FreeBSD ports tree.

Introducing the FreeBSD Ports Tree

Thus far we have worked with FreeBSD packages. They are convenient, but they do not independently support an update mechanism. The reference against which packages are compared to determine their *freshness* is the FreeBSD ports tree. On our reference FreeBSD 7.1 system, we installed the version of the ports tree that shipped with FreeBSD 7.1 RELEASE.

The FreeBSD ports tree can be found in the `/usr/ports` directory (see Listing 8). For the purposes of this article, it is sufficient to know that FreeBSD ports are a framework upon which application source code is installed on a FreeBSD system.

Updating the FreeBSD Ports Tree

The easiest way to update the FreeBSD ports tree is to use Colin Percival's Portsnap tool (*http://www.daemonology.net/portsnap/*), now shipped with FreeBSD. First run `portsnap fetch` to download a compressed version of the FreeBSD ports tree needed by portsnap (see Listing 9).

In the future, we do not need to run `portsnap extract`. Instead, we run `portsnap update`.

With the FreeBSD ports tree installed, we can use the `pkg_version` tool to check what packages need to be updated. This checks for any update, not just security updates as we saw with Portaudit (see Listing 10).

As we can see, two of our packages (Curl and Portaudit) have newer versions available.

Reading /usr/ports/UPDATING

It is important to read `/usr/ports/UPDATING` before invoking Portupgrade. We have not done so yet because these examples have

Listing 4. pub/FreeBSD/ports/i386 directory listing

```
Oct 23  2006  Symbolic Link    packages -> packages-stable
Aug 24 12:57  Directory        packages-6-stable
Nov 21  2008  Directory        packages-6.4-release
Aug 22 21:56  Directory        packages-7-stable
Dec 22  2008  Directory        packages-7.1-release
May  1 18:42  Directory        packages-7.2-release
Aug 20 21:36  Symbolic Link    packages-8-current -> packages-8-stable
Aug 24 13:05  Directory        packages-8-stable
Aug 18 00:32  Directory        packages-9-current
Feb  9  2008  Symbolic Link    packages-current -> packages-8-current/
Mar  1  2008  Symbolic Link    packages-stable -> packages-7-stable
```

Listing 5. Removing a FreeBSD package with pkg_delete

```
freebsd7# pkg_info
curl-7.18.0         Non-interactive tool to get files from FTP, GOPHER,
HTTP(S)
portaudit-0.5.12    Checks installed ports against a list of security
vulnerabi
freebsd7# cd /var/db/pkg/
freebsd7# ls
curl-7.18.0
freebsd7# pkg_delete curl-7.18.0/
freebsd7# pkg_info
portaudit-0.5.12    Checks installed ports against a list of security
vulnerabi
```

Listing 6. Installing a newer curl package using pkg_add

```
freebsd7# setenv PACKAGESITE ftp://ftp.freebsd.org//pub/
FreeBSD/ports/i386/packages-7.2-release/Latest/
freebsd7# pkg_add -vr curl
scheme:   [ftp]
user:     []
password: []
host:     [ftp.freebsd.org]
port:     [0]
document: [//pub/FreeBSD/ports/i386/packages-7.2-
release/Latest/curl.tbz]
scheme:   [http]
user:     []
password: []
host:     [172.16.2.1]
port:     [3128]
document: [/]
---> 172.16.2.1:3128
looking up 172.16.2.1
connecting to 172.16.2.1:3128
requesting ftp://ftp.freebsd.org//pub/FreeBSD/ports/
i386/packages-7.2-release/Latest/curl.tbz
>>> GET ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7.2-release/Latest/curl.tbz HTTP/1.1
>>> Host: ftp.freebsd.org
>>> User-Agent: pkg_add libfetch/2.0
>>> Connection: close
>>>
<<< HTTP/1.0 200 Gatewaying
<<< Server: squid/2.7.STABLE6
<<< Date: Mon, 24 Aug 2009 20:00:58 GMT
<<< Content-Type: text/plain
<<< Content-Length: 1097934
content length: [1097934]
<<< Last-Modified: Mon, 13 Apr 2009 21:18:46 GMT
last modified: [2009-04-13 21:18:46]
<<< X-Cache: MISS from r200a.taosecurity.com
<<< Via: 1.0 r200a.taosecurity.com:3128 (squid/
2.7.STABLE6)
<<< Connection: close
<<<
offset 0, length -1, size -1, clength 1097934
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7.2-release/Latest/curl.tbz...
x +CONTENTS
x +COMMENT
x +DESC
x +MTREE_DIRS
x man/man1/curl.1.gz
...edited...
x share/examples/curl/threaded-ssl.c
tar command returns 0 status
 Done.
Package 'curl-7.19.4' depends on 'ca_root_nss-3.11.9_2'
with 'security/ca_root_nss' origin.
scheme:   [ftp]
user:     []
password: []
host:     [ftp.freebsd.org]
port:     [0]
document: [//pub/FreeBSD/ports/i386/packages-7.2-
release/All/ca_root_nss-3.11.9_2.tbz]
scheme:   [http]
user:     []
password: []
host:     [172.16.2.1]
port:     [3128]
document: [/]
---> 172.16.2.1:3128
looking up 172.16.2.1
connecting to 172.16.2.1:3128
requesting ftp://ftp.freebsd.org//pub/FreeBSD/ports/
i386/packages-7.2-release/All/ca_root_nss-3.11.9_2.tbz
>>> GET ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7.2-release/All/ca_root_nss-3.11.9_2.tbz HTTP/1.1
>>> Host: ftp.freebsd.org
>>> User-Agent: pkg_add libfetch/2.0
>>> Connection: close
>>>
<<< HTTP/1.0 200 Gatewaying
<<< Server: squid/2.7.STABLE6
<<< Date: Mon, 24 Aug 2009 20:01:02 GMT
<<< Content-Type: text/plain
<<< Content-Length: 172602
content length: [172602]
<<< Last-Modified: Mon, 13 Apr 2009 21:00:07 GMT
last modified: [2009-04-13 21:00:07]
<<< X-Cache: MISS from r200a.taosecurity.com
<<< Via: 1.0 r200a.taosecurity.com:3128 (squid/
2.7.STABLE6)
<<< Connection: close
<<<
offset 0, length -1, size -1, clength 172602
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7.2-release/All/ca_root_nss-3.11.9_2.tbz...
x +CONTENTS
x +COMMENT
x +DESC
x +MTREE_DIRS
x share/certs/ca-root-nss.crt
tar command returns 0 status
 Done.
Finished loading ca_root_nss-3.11.9_2 over FTP.
extract: Package name is ca_root_nss-3.11.9_2
extract: CWD to /usr/local
```

HAKIN9

PRACTICAL PROTECTION

SAINT® Integrated Vulnerability and Penetration Testing

saintcorporation.com

HAKIN9

PRACTICAL PROTECTION HARD CORE IT SECURITY MAGAZINE

HARDWARE KEYLOGGER A SERIOUS THREAT

HOW TO STAY SAFE

CWNA TRAINING

THE FEAR FACTOR – STUDY OF A NEW GENRE OF MALWARES CALLED "SCAREWARES"

DEBUGGERS – KNOW YOUR ENEMY

SMS TRICKERY IN PUBLIC TRANSPORT

FILE CARVING – TERMINOLOGY, BASICS AND TOOLS

BEHAVIORAL ANALYSIS OF UNWISE .EXE MALWARE

APPLICATIONS ON THE CD

CERTIFIED WIRELESS NETWORK ADMINISTRATOR TRAINING BY SECURITORG

AD-AWARE PRO 4.29 BY LAVASOFT

PLUS

WINDOWS TIMELINE ANALYSIS, PART 3 (ADVANCED & ALTERNATIVE TOPICS) BY HARLAN CARVEY

APC

T YOUR COMPUTER, IENT, AND YOUR WALLET

IT SECURITY MAGAZINE

WWW.HAKIN9.ORG/EN

Listing 6. Installing a newer curl package using pkg_add (cont)

```
extract: /usr/local/share/certs/ca-root-nss.crt
extract: CWD to .
Running mtree for ca_root_nss-3.11.9_2..
mtree -U -f +MTREE_DIRS -d -e -p /usr/local >/dev/null
Attempting to record package into /var/db/pkg/ca_root_nss-3.11.9_2..
Package ca_root_nss-3.11.9_2 registered in /var/db/pkg/ca_root_nss-3.11.9_2
   'ca_root_nss-3.11.9_2' loaded successfully.
extract: Package name is curl-7.19.4
extract: CWD to /usr/local
extract: /usr/local/man/man1/curl.1.gz
...edited...
extract: /usr/local/share/examples/curl/threaded-ssl.c
extract: execute '/sbin/ldconfig -m /usr/local/lib'
extract: CWD to .
Running mtree for curl-7.19.4..
mtree -U -f +MTREE_DIRS -d -e -p /usr/local >/dev/null
Attempting to record package into /var/db/pkg/curl-7.19.4..
Trying to record dependency on package 'ca_root_nss-3.11.9_2' with 'security/ca_root_nss' origin.
Package curl-7.19.4 registered in /var/db/pkg/curl-7.19.4
freebsd7# pkg_info
ca_root_nss-3.11.9_2 The root certificate bundle from the Mozilla Project
curl-7.19.4          Non-interactive tool to get files from FTP, GOPHER, HTTP(S)
portaudit-0.5.12     Checks installed ports against a list of security vulnerabi
```

Listing 7. Portaudit output shows curl is no longer vulnerable

```
freebsd7# portaudit -Fdav
Attempting to fetch from http://www.FreeBSD.org/ports/.
auditfile.tbz                                 100% of   57 kB   69 kBps
New database installed.
Database created: Mon Aug 24 15:40:01 EDT 2009
0 problem(s) in your installed packages found.
```

Listing 8. FreeBSD ports tree

```
freebsd7# ls /usr/ports
.cvsignore      arabic         emulators     mbone          shells
CHANGES         archivers      finance       misc           sysutils
COPYRIGHT       astro          french        multimedia     textproc
GIDs            audio          ftp           net            ukrainian
INDEX-7         benchmarks     games         net-im         vietnamese
KNOBS           biology        german        net-mgmt       www
LEGAL           cad            graphics      net-p2p        x11
MOVED           chinese        hebrew        news           x11-clocks
Makefile        comms          hungarian     palm           x11-drivers
Mk              converters     irc           polish         x11-fm
README          databases     japanese       ports-mgmt     x11-fonts
Templates       deskutils     java           portuguese     x11-servers
Tools           devel         korean         print          x11-themes
UIDs            distfiles     langTY         russian        x11-toolkits
UPDATING        dns           mail           science        x11-wm
accessibility   editors       math           security
```

been fairly simple. However, there may be information in /usr/ports/UPDATING that could recommend different actions depending on the ports of interest. From now on, consult /usr/ports/UPDATING after you upgrade your ports tree and before you invoke Portupgrade.

Installing Portupgrade

The FreeBSD Portupgrade program (*http://wiki.freebsd.org/portupgrade*) is a powerful tool that offers the ability to update packages using only packages. Portupgrade is a bit *heavy* in the sense that it requires installing Ruby as a dependency, whereas other options do not require such dependencies. However, other options do not seem to have the ability to dictate installing packages instead of building from ports.

We'll install Portupgrade from the 7-stable package collection by setting the appropriate environment variable and then invoking pkg_add. When we start we have only 3 packages installed (see Listing 11).

After adding Portupgrade, we have 7 packages installed. You can see the Ruby and Berkeley DB dependencies installed by Portupgrade.

Updating Packages Using Portupgrade

With Portupgrade installed, we can use the portversion tool to determine what packages need updating (see Listing 12). If we just want to see packages that need updating, we run portversion -v -l "<".

When we run portversion, we see it builds a package database (pkgdb), then a ports database (portsdb) for its own use. We could have used pkg_version to produce the same output (see Listing 13).

So, with this information, how can we update packages that need updating?

The following advice is based on my personal preferences, but when updating packages I prefer to use packages, not compiling from source code, when possible. (I'll discuss alternatives later.) The following example will update the packages for which newer version are available.

First we set proxy and PACKAGESITE variables, and then we invoke Portupgrade (see Listing 14).

When done, we can see the packages have been updated (see Listing 15).

So what just happened? Portupgrade found that Curl and Portaudit were out-of-date. It downloaded the newest packages from the packages-7-stable directory on a remote FreeBSD FTP server, uninstalled the out-of-date package, and installed the up-to-date package.

If you noticed in the Portupgrade output, the program stores copies of the packages it downloads in the /usr/ports/packages/All directory.

```
freebsd7# ls /usr/ports/packages/All
curl-7.19.6_1.tbz    portaudit-0.5.13.tbz
```

By specifying the -a switch we told Portupgrade to update all packages. The -v switch enabled verbose mode. The -PP switch told Portupgrade to only use packages, and it retrieved those packages from the public FreeBSD package repository.

There are other ways to invoke Portupgrade, such as telling it to only update individual packages, and then update their dependencies, and so on. I prefer this simpler approach of updating everything that requires it.

FreeBSD Package Dependencies

Dependencies are packages which are required in order to run other packages. We can use the pkg_info command to

Listing 9. Updating the ports tree with portsnap

```
freebsd7# portsnap fetch
Looking up portsnap.FreeBSD.org mirrors... 3 mirrors found.
Fetching public key from portsnap2.FreeBSD.org... done.
Fetching snapshot tag from portsnap2.FreeBSD.org... done.
Fetching snapshot metadata... done.
Fetching snapshot generated at Sun Aug 23 20:41:07 EDT 2009:
e4a063906c569a6d82cdc053dda2ced013f53d80723ef4100% of   59 MB  359 kBps
00m00s
Extracting snapshot... done.
Verifying snapshot integrity... done.
Fetching snapshot tag from portsnap2.FreeBSD.org... done.
Fetching snapshot metadata... done.
Updating from Sun Aug 23 20:41:07 EDT 2009 to Mon Aug 24 13:52:56 EDT 2009.
Fetching 4 metadata patches... done.
Applying metadata patches... done.
Fetching 0 metadata files... done.
Fetching 36 patches.....10....20....30... done.
Applying patches... done.
Fetching 2 new ports or files... done.

freebsd7# portsnap extract
/usr/ports/.cvsignore
/usr/ports/CHANGES
/usr/ports/COPYRIGHT
...edited...
/usr/ports/x11/zenity/
Building new INDEX files... done.
```

Listing 10. Using pkg_version to check for updates

```
freebsd7# pkg_version -v
ca_root_nss-3.11.9_2               =    up-to-date with port
curl-7.19.4                        <    needs updating (port has 7.19.6_1)
portaudit-0.5.12                   <    needs updating (port has 0.5.13)
```

Listing 11. Installing portupgrade using pkg_add

```
freebsd7# pkg_info
ca_root_nss-3.11.9_2 The root certificate bundle from the
Mozilla Project
curl-7.19.4          Non-interactive tool to get files
from FTP, GOPHER, HTTP(S)
portaudit-0.5.12     Checks installed ports against a
list of security vulnerabi
freebsd7# setenv PACKAGESITE ftp://ftp.freebsd.org//pub/
FreeBSD/ports/i386/packages-7-stable/Latest/
freebsd7# pkg_add -vr portupgrade
scheme:   [ftp]
user:     []
password: []
host:     [ftp.freebsd.org]
port:     [0]
document: [//pub/FreeBSD/ports/i386/packages-7-stable/
Latest/portupgrade.tbz]
scheme:   [http]
user:     []
password: []
host:     [172.16.2.1]
port:     [3128]
document: [/]
---> 172.16.2.1:3128
looking up 172.16.2.1
connecting to 172.16.2.1:3128
requesting ftp://ftp.freebsd.org//pub/FreeBSD/ports/
i386/packages-7-stable/Latest/portupgrade.tbz
>>> GET ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/Latest/portupgrade.tbz HTTP/1.1
>>> Host: ftp.freebsd.org
>>> User-Agent: pkg_add libfetch/2.0
>>> Connection: close
...edited...
Running mtree for portupgrade-2.4.6_3,2..
mtree -U -f +MTREE_DIRS -d -e -p /usr/local >/dev/null
Attempting to record package into /var/db/pkg/
portupgrade-2.4.6_3,2..
Trying to record dependency on package 'ruby-1.8.7.160_
4,1' with 'lang/ruby18' origin.
Trying to record dependency on package 'db41-4.1.25_4'
with 'databases/db41' origin.
Trying to record dependency on package 'ruby18-bdb-
0.6.5_1' with 'databases/ruby-bdb' origin.
Package portupgrade-2.4.6_3,2 registered in /var/db/pkg/
portupgrade-2.4.6_3,2
------------------------------------------------------
      Fill ALT_PKGDEP section in pkgtools.conf file for
portupgrade
      be aware of alternative dependencies you use.
      E.g.
      ALT_PKGDEP = {
            'www/apache13' => 'www/apache13-modssl',
      'print/ghostscript-gnu' => 'print/ghostscript-gpl',
      }
      Note also, portupgrade knows nothing how to handle
ports with different
      suffixes (E.g. -nox11). So you should explicitly
define variables
      (E.g. WITHOUT_X11=yes) for the ports in /etc/
make.conf or pkgtools.conf
      (MAKE_ARGS section) files.
------------------------------------------------------
freebsd7# rehash
freebsd7# pkg_info
ca_root_nss-3.11.9_2 The root certificate bundle from the
Mozilla Project
curl-7.19.4          Non-interactive tool to get files
from FTP, GOPHER, HTTP(S)
db41-4.1.25_4        The Berkeley DB package, revision
4.1
portaudit-0.5.12     Checks installed ports against a
list of security vulnerabi
portupgrade-2.4.6_3,2 FreeBSD ports/packages
administration and management tool s
ruby-1.8.7.160_4,1  An object-oriented interpreted
scripting language
ruby18-bdb-0.6.5_1  Ruby interface to Sleepycat's
Berkeley DB revision 2 or lat
```

Listing 12. Using portversion to check for updates

```
freebsd7# portversion -v
[Rebuilding the pkgdb <format:bdb_btree> in /var/db/pkg
... - 7 packages found (-0 +7) ....... done]
[Updating the portsdb <format:bdb_btree> in /usr/ports
... - 20616 port entries found .........1000.........200
0.........3000.........4000.........5000.........6000
.........7000.........8000.........9000.........10000...
......11000.........12000
.........13000.........14000.........15000.........16000
.........17000.........18000
.........19000.........20000...... ..... done]

ca_root_nss-3.11.9_2        =  up-to-date with port
curl-7.19.4                 <  needs updating (port has
7.19.6_1)
db41-4.1.25_4               =  up-to-date with port
portaudit-0.5.12            <  needs updating (port has
0.5.13)
portupgrade-2.4.6_3,2       =  up-to-date with port
ruby-1.8.7.160_4,1          =  up-to-date with port
ruby18-bdb-0.6.5_1          =  up-to-date with port
```

Listing 13. Using pkg_version to check for updates

```
freebsd7# pkg_version -v
ca_root_nss-3.11.9_2    = up-to-date with port
curl-7.19.4            < needs updating (port has 7.19.6_1)
db41-4.1.25_4           = up-to-date with port
portaudit-0.5.12        < needs updating (port has 0.5.13)
portupgrade-2.4.6_3,2   = up-to-date with port
ruby-1.8.7.160_4,       = up-to-date with port
ruby18-bdb-0.6.5_1      = up-to-date with port
```

Listing 14. Updating old packages with new packages using portupgrade

```
freebsd7# setenv HTTP_PROXY http://172.16.2.1:3128
freebsd7# setenv PACKAGESITE ftp://ftp.freebsd.org//pub/
FreeBSD/ports/i386/packages-7-stable/Latest/
freebsd7# portupgrade -vaPP
--->  Session started at: Tue, 25 Aug 2009 09:28:59 -0400
--->  Checking for the latest package of 'ftp/curl'
** No such file or directory - /usr/ports/packages/All
--->  Fetching the package(s) for 'curl-7.19.6_1' (ftp/curl)
--->  Fetching curl-7.19.6_1
++ Will try the following sites in the order named:
    ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-7-stable/
--->  Invoking a command: /usr/bin/fetch -o '/var/
tmp/portupgradeRvOuj4wl/curl-7.19.6_1.tbz' 'ftp://
ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-7-stable/
All/curl-7.19.6_1.tbz'
/var/tmp/portupgradeRvOuj4wl/curl-7.19.6_1.tbz100% of 1103
kB 2824 kBps
--->  Downloaded as curl-7.19.6_1.tbz
--->  Identifying the package /var/tmp/
portupgradeRvOuj4wl/curl-7.19.6_1.tbz
--->  Saved as /usr/ports/packages/All/curl-7.19.6_1.tbz
--->  Listing the results (+:done / -:ignored / *:skipped / !:failed)
    + curl-7.19.6_1
--->  Packages processed: 1 done, 0 ignored, 0 skipped and 0 failed
--->  Found a package of 'ftp/curl': /usr/ports/packages/
All/curl-7.19.6_1.tbz (curl-7.19.6_1)
--->  Located a package version 7.19.6_1 (/usr/ports/
packages/All/curl-7.19.6_1.tbz)
--->  Upgrade of ftp/curl started at: Tue, 25 Aug 2009 09:29:18 -0400
--->  Upgrading 'curl-7.19.4' to 'curl-7.19.6_1' (ftp/
curl) using a package
--->  Updating dependency info
--->  Uninstallation of curl-7.19.4 started at: Tue, 25
Aug 2009 09:29:18 -0400
--->  Fixing up dependencies before creating a package
--->  Backing up the old version
--->  Uninstalling the old version
--->  Deinstalling 'curl-7.19.4'
--->  Preserving /usr/local/lib/libcurl.so.5 as /usr/
local/lib/compat/pkg/libcurl.so.5
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ...
- 6 packages found (-1 +0) (...) done]
--->  Uninstallation of curl-7.19.4 ended at: Tue, 25 Aug
2009 09:29:34 -0400 (consumed 00:00:15)
--->  Installation of curl-7.19.6_1 started at: Tue, 25
Aug 2009 09:29:34 -0400
--->  Installing the new version via the package
--->  Removing temporary files and directories
--->  Removing old package'
--->  Installation of curl-7.19.6_1 ended at: Tue, 25 Aug
2009 09:29:38 -0400 (consumed 00:00:04)
--->  Cleaning out obsolete shared libraries
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ...
- 7 packages found (-0 +1) . done]
--->  Upgrade of ftp/curl ended at: Tue, 25 Aug 2009 09:
29:49 -0400 (consumed 00:00:30)
--->  ** Upgrade tasks 2: 1 done, 0 ignored, 0 skipped and
0 failed
--->  Checking for the latest package of 'ports-mgmt/
portaudit'
--->  Fetching the package(s) for 'portaudit-0.5.13'
(ports-mgmt/portaudit)
--->  Fetching portaudit-0.5.13
++ Will try the following sites in the order named:
    ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-
7-stable/
--->  Invoking a command: /usr/bin/fetch -o '/var/tmp/
portupgradeY8v1o54H/portaudit-0.5.13.tbz' 'ftp://
ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-7-stable/
All/portaudit-0.5.13.tbz'
/var/tmp/portupgradeY8v1o54H/portaudit-0.5.13.100% of   10
kB 1842 kBps
--->  Downloaded as portaudit-0.5.13.tbz
--->  Identifying the package /var/tmp/portupgradeY8v1o54H/
portaudit-0.5.13.tbz
--->  Saved as /usr/ports/packages/All/portaudit-
0.5.13.tbz
--->  Listing the results (+:done / -:ignored / *:skipped / !:
failed)
    + portaudit-0.5.13
--->  Packages processed: 1 done, 0 ignored, 0 skipped and
0 failed
--->  Found a package of 'ports-mgmt/portaudit': /usr/ports/
packages/All/portaudit-0.5.13.tbz (portaudit-0.5.13)
--->  Located a package version 0.5.13 (/usr/ports/
packages/All/portaudit-0.5.13.tbz)
--->  Upgrade of ports-mgmt/portaudit started at: Tue, 25
Aug 2009 09:29:59 -0400
--->  Upgrading 'portaudit-0.5.12' to 'portaudit-0.5.13'
(ports-mgmt/portaudit) using a package
--->  Updating dependency info
--->  Uninstallation of portaudit-0.5.12 started at: Tue,
25 Aug 2009 09:30:00 -0400
--->  Fixing up dependencies before creating a package
```

Listing 14. Updating old packages with new packages using portupgrade (cont)

```
--->  Backing up the old version
--->  Uninstalling the old version
--->  Deinstalling 'portaudit-0.5.12'
The portaudit package has been deleted.
If you're *not* upgrading and won't be using
it any longer, you may want to remove the portaudit database:
  rm -Rf /var/db/portaudit
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ...
- 6 packages found (-1 +0) (...) done]
--->  Uninstallation of portaudit-0.5.12 ended at: Tue, 25
Aug 2009 09:30:20 -0400 (consumed 00:00:20)
--->  Installation of portaudit-0.5.13 started at: Tue, 25
Aug 2009 09:30:20 -0400
--->  Installing the new version via the package
--->  Removing temporary files and directories
--->  Removing old package'
--->  Installation of portaudit-0.5.13 ended at: Tue, 25
Aug 2009 09:30:22 -0400 (consumed 00:00:01)
--->  Cleaning out obsolete shared libraries
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ...
- 7 packages found (-0 +1) . done]
--->  Upgrade of ports-mgmt/portaudit ended at: Tue, 25
Aug 2009 09:30:31 -0400 (consumed 00:00:32)
--->  ** Upgrade tasks 2: 2 done, 0 ignored, 0 skipped and 0 failed
--->  Listing the results (+:done / -:ignored / *:skipped / !:failed)
   + ftp/curl (curl-7.19.4)
   + ports-mgmt/portaudit (portaudit-0.5.12)
--->  Packages processed: 2 done, 0 ignored, 0 skipped and 0 failed
--->  Session ended at: Tue, 25 Aug 2009 09:30:40 -0400
(consumed 00:01:40)
```

Listing 15. Portversion shows packages as up-to-date

```
freebsd7# portversion -v
ca_root_nss-3.11.9_2          =  up-to-date with port
curl-7.19.6_1                 =  up-to-date with port
db41-4.1.25_4                 =  up-to-date with port
portaudit-0.5.13              =  up-to-date with port
portupgrade-2.4.6_3,2         =  up-to-date with port
ruby-1.8.7.160_4,1            =  up-to-date with port
ruby18-bdb-0.6.5_1            =  up-to-date with port
```

Listing 16. Installing pkg_tree

```
freebsd7# pkg_add -r pkg_tree
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/Latest/pkg_tree.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/All/perl-5.8.9_3.tbz... Done.
Removing stale symlinks from /usr/bin...
    Skipping /usr/bin/perl
    Skipping /usr/bin/perl5
Done.
Creating various symlinks in /usr/bin...
    Symlinking /usr/local/bin/perl5.8.9 to /usr/bin/perl
    Symlinking /usr/local/bin/perl5.8.9 to /usr/bin/perl5
Done.
Cleaning up /etc/make.conf... Done.
Spamming /etc/make.conf... Done.
Cleaning up /etc/manpath.config... Done.
Spamming /etc/manpath.config... Done.
```

Listing 17. Running pkg_tree

```
freebsd7# pkg_tree
ca_root_nss-3.11.9_2
curl-7.19.6_1
 \__ ca_root_nss-3.11.9_2
db41-4.1.25_4
perl-5.8.9_3
pkg_tree-1.1_1
 \__ perl-5.8.9_3
portaudit-0.5.13
portupgrade-2.4.6_3,2
|\__ ruby-1.8.7.160_4,1
|\__ db41-4.1.25_4
 \__ ruby18-bdb-0.6.5_1
ruby-1.8.7.160_4,1
ruby18-bdb-0.6.5_1
|\__ ruby-1.8.7.160_4,1
 \__ db41-4.1.25_4
```

Listing 18. Installing mutt and updating pkg_db

```
freebsd7# pkg_add -r mutt
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/Latest/mutt.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/All/urlview-0.9_2.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/All/ispell-3.3.02_4.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/All/mime-support-3.46.1.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/All/png-1.2.38.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/All/pcre-7.9.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/All/libiconv-1.13.1.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/All/libslang2-2.1.4_1.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/
packages-7-stable/All/gettext-0.17_1.tbz... Done.
freebsd7# pkgdb -vu
--->  Updating the pkgdb
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ...
- 18 packages found (-0 +9) ........ done]
```

Open Source Days

Copenhagen

05-06 MARCH

www.opensourcedays.org

Listing 19. Failing to remove pcre because of dependencies

```
freebsd7# pkg_deinstall pcre-7.9/
--->  Deinstalling 'pcre-7.9'
pkg_delete: package 'pcre-7.9' is required by these other packages
and may not be deinstalled:
libslang2-2.1.4_1
mutt-1.4.2.3_3
** Listing the failed packages (-:ignored / *:skipped / !:failed)
   ! pcre-7.9    (pkg_delete failed)
```

Listing 20. Removing mutt and its dependencies using pkg_deinstall

```
freebsd7# pkg_deinstall -R mutt-1.4.2.3_3/
--->  Deinstalling 'mutt-1.4.2.3_3'
--->  Deinstalling 'urlview-0.9_2'
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 17 packages found (-1 +0) (...) done]
--->  Deinstalling 'libslang2-2.1.4_1'
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 16 packages found (-1 +0) (...) done]
--->  Deinstalling 'pcre-7.9'
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 15 packages found (-1 +0) (...) done]
--->  Deinstalling 'mime-support-3.46.1'
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 14 packages found (-1 +0) (...) done]
--->  Deinstalling 'png-1.2.38'
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 13 packages found (-1 +0) (...) done]
--->  Deinstalling 'gettext-0.17_1'
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 12 packages found (-1 +0) (...) done]
--->  Deinstalling 'libiconv-1.13.1'
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 11 packages found (-1 +0) (...) done]
--->  Deinstalling 'ispell-3.3.02_4'
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 10 packages found (-1 +0) (...) done]
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 9 packages found (-1 +0) (...) done]
```

Listing 21. Installning nmap and tcpflow

```
freebsd7# pkg_add -r nmap
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/packages-7-stable/Latest/nmap.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/packages-7-stable/All/pkg-config-0.23_1.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/packages-7-stable/All/lua-5.1.4.tbz... Done.
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/packages-7-stable/All/pcre-7.9.tbz... Done.
freebsd7# pkg_add -r tcpflow
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/packages-7-stable/Latest/tcpflow.tbz... Done.
freebsd7# pkgdb -vu
--->  Updating the pkgdb
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 14 packages found (-0 +5) ..... done]
```

Listing 22. Installing pkg_cutleaves

```
freebsd7# pkg_add -r pkg_cutleaves
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/packages-7-stable/Latest/pkg_cutleaves.tbz... Done.
freebsd7# pkgdb -vu
--->  Updating the pkgdb
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 15 packages found (-0 +1) . done]
freebsd7# rehash
```

learn what packages a specified package depends on.

```
freebsd7# pkg_info -rx curl
Information for curl-7.19.6_1:

Depends on:
Dependency: ca_root_nss-3.11.9_2
```

Here we see that curl depends on the `ca_root_nss` package. The `-r` command tells `pkg_info` to display the packages on which curl depends. The `-x` switch tells `pkg_info` to do a regular expression match, so we don't have to list the whole package name. Does anything depend on curl?

Listing 23. Using pkg_cutleaves to remove nmap and tcpflow and dependencies

```
freebsd7# pkg_cutleaves
Package 1 of 7:
curl-7.19.6_1 - Non-interactive tool to get files from
FTP, GOPHER, HTTP(S) servers
curl-7.19.6_1 - [keep]/(d)elete/(f)lush marked pkgs/
(a)bort?
** Keeping curl-7.19.6_1.

Package 2 of 7:
nmap-5.00 - Port scanning utility for large networks
nmap-5.00 - [keep]/(d)elete/(f)lush marked pkgs/(a)bort?
d
** Marking nmap-5.00 for removal.

Package 3 of 7:
pkg_cutleaves-20090810 - Interactive script for
deinstalling 'leaf' packages
pkg_cutleaves-20090810 - [keep]/(d)elete/(f)lush marked
pkgs/(a)bort?
** Keeping pkg_cutleaves-20090810.

Package 4 of 7:
pkg_tree-1.1_1 - Get a 'graphical' tree-overview of
installed packages
pkg_tree-1.1_1 - [keep]/(d)elete/(f)lush marked pkgs/
(a)bort?
** Keeping pkg_tree-1.1_1.

Package 5 of 7:
portaudit-0.5.13 - Checks installed ports against a list
of security vulnerabilities
portaudit-0.5.13 - [keep]/(d)elete/(f)lush marked pkgs/
(a)bort?
** Keeping portaudit-0.5.13.

Package 6 of 7:
portupgrade-2.4.6_3,2 - FreeBSD ports/packages
administration and management tool suite
portupgrade-2.4.6_3,2 - [keep]/(d)elete/(f)lush marked
pkgs/(a)bort?
** Keeping portupgrade-2.4.6_3,2.

Package 7 of 7:
tcpflow-0.21_1 - A tool for capturing data transmitted as
part of TCP connections
tcpflow-0.21_1 - [keep]/(d)elete/(f)lush marked pkgs/
(a)bort? d
** Marking tcpflow-0.21_1 for removal.

Deleting nmap-5.00 (package 1 of 2).
Deleting tcpflow-0.21_1 (package 2 of 2).
Go on with new leaf packages ([yes]/no)? y

Package 1 of 2:
lua-5.1.4 - Small, compilable scripting language
providing easy access to C code
lua-5.1.4 - [keep]/(d)elete/(f)lush marked pkgs/(a)bort?
d
** Marking lua-5.1.4 for removal.

Package 2 of 2:
pcre-7.9 - Perl Compatible Regular Expressions library
pcre-7.9 - [keep]/(d)elete/(f)lush marked pkgs/(a)bort?
d
** Marking pcre-7.9 for removal.

Deleting lua-5.1.4 (package 1 of 2).
Deleting pcre-7.9 (package 2 of 2).
Go on with new leaf packages ([yes]/no)? y

Package 1 of 1:
pkg-config-0.23_1 - A utility to retrieve information
about installed libraries
pkg-config-0.23_1 - [keep]/(d)elete/(f)lush marked pkgs/
(a)bort? d
** Marking pkg-config-0.23_1 for removal.

Deleting pkg-config-0.23_1 (package 1 of 1).
** Didn't find any new leaves to work with, exiting.
** Deinstalled packages:
lua-5.1.4
nmap-5.00
pcre-7.9
pkg-config-0.23_1
tcpflow-0.21_1
** Number of deinstalled packages: 5

freebsd7# pkgdb -vu
--->  Updating the pkgdb
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg
... - 10 packages found (-5 +0) (...) done]
```

Listing 24. Updating the ports tree with portsnap

```
freebsd7# portsnap fetch
Looking up portsnap.FreeBSD.org mirrors... 3 mirrors found.
Fetching snapshot tag from portsnap2.FreeBSD.org... done.
Fetching snapshot metadata... done.
Updating from Mon Aug 24 13:52:56 EDT 2009 to Tue Aug 25 07:32:23 EDT 2009.
Fetching 4 metadata patches... done.
Applying metadata patches... done.
Fetching 0 metadata files... done.
Fetching 32 patches.....10....20....30. done.
Applying patches... done.
Fetching 2 new ports or files... done.

freebsd7# portsnap update
Removing old files and directories... done.
Extracting new files:
/usr/ports/audio/gtkpod/
/usr/ports/databases/pgadmin3/
/usr/ports/devel/cvsnt/
/usr/ports/devel/git/
/usr/ports/devel/jude-community/
/usr/ports/devel/p5-local-lib/
/usr/ports/games/wesnoth/
/usr/ports/graphics/Makefile
/usr/ports/graphics/mmrecover/
/usr/ports/graphics/rubygem-scruffy/
/usr/ports/mail/meta1/
/usr/ports/net-mgmt/nagios-plugins/
/usr/ports/net/nss_ldapd/
/usr/ports/ports-mgmt/portmaster/
/usr/ports/security/fiked/
/usr/ports/security/swatch/
/usr/ports/security/vuxml/
/usr/ports/sysutils/e2fsprogs/
/usr/ports/sysutils/libchk/
/usr/ports/sysutils/virtualmin/
/usr/ports/textproc/ansifilter/
/usr/ports/textproc/yodl/
/usr/ports/www/Makefile
/usr/ports/www/apache22/
/usr/ports/www/elinks/
/usr/ports/www/galeon/
/usr/ports/www/gist/
/usr/ports/www/p5-Catalyst-View-JSON/
/usr/ports/www/p5-HTTP-Engine/
/usr/ports/www/pyweblib/
/usr/ports/x11-clocks/xdaliclock/
/usr/ports/x11/gdm/
/usr/ports/x11/gnome2-fifth-toe/
/usr/ports/x11/xorg/
Building new INDEX files... done.
```

```
freebsd7# pkg_info -Rx curl
Information for curl-7.19.6_1:
```

The `-R` switch shows that nothing depends on curl. If we ran this command for `ca_root_nss`, however, we would see that curl requires it.

```
freebsd7# pkg_info -Rx ca_root_nss
Information for ca_root_nss-3.11.9_2:
Required by:
curl-7.19.6_1
```

Another way to understand these relationships is to install the `pkg_tree` package. From now on, when adding new packages, it helps to update the package database maintained by Portupgrade, using the `pkgdb` command.

```
freebsd7# pkgdb -vu
--->  Updating the pkgdb
[Updating the pkgdb <format:bdb_btree>
in /var/db/pkg ... - 9 packages found
(-0 +2) .. done]
```

If you forget to run pkgdb after installing a package, it's not a big problem. Any time a tool in the Portupgrade suite is invoked (such as portupgrade itself, or other tools), the pkgdb will be updated. During the `pkg_tree` installation process we saw Perl installed as a dependency of `pkg_tree`. Once installed, run `pkg_tree` and tell it to show what packages curl depends on.

```
freebsd7# pkg_tree curl
curl-7.19.6_1
 \__ ca_root_nss-3.11.9_2
```

Portupgrade presents a more complicated example.

```
freebsd7# pkg_tree portupgrade
portupgrade-2.4.6_3,2
|\__ ruby-1.8.7.160_4,1
|\__ db41-4.1.25_4
 \__ ruby18-bdb-0.6.5_1
```

We can go one step farther to follow the dependency chain using the `-v` switch.

```
freebsd7# pkg_tree -v portupgrade
portupgrade-2.4.6_3,2
|\__ ruby-1.8.7.160_4,1
|\__ db41-4.1.25_4
```

```
 \__ ruby18-bdb-0.6.5_1
      |\__ ruby-1.8.7.160_4,1
       \__ db41-4.1.25_4
```

Now we see that Portupgrade depends on ruby, db41, and ruby18-bdb. However, ruby18-bdb depends on ruby and db41 as well. Running `pkg_tree` with no options shows all package dependencies (see Listing 17). Understanding dependencies is important, because FreeBSD won't let you delete a package when another package depends on it. We'll look at that next.

Removing Packages

For the following examples we add the open source text email client Mutt to our system. When you check Mutt's dependencies, you find several:

```
freebsd7# pkg_tree mutt
mutt-1.4.2.3_3
|\__ urlview-0.9_2
|\__ ispell-3.3.02_4
|\__ mime-support-3.46.1
|\__ png-1.2.38
|\__ pcre-7.9
|\__ libiconv-1.13.1
|\__ libslang2-2.1.4_1
 \__ gettext-0.17_1
```

If you try to delete, say, the pcre package, the attempt will fail.

```
freebsd7# pkg_delete pcre-7.9/
pkg_delete: package 'pcre-7.9' is
required by these other packages
and may not be deinstalled:
libslang2-2.1.4_1
mutt-1.4.2.3_3
```

If you try using the `pkg_deinstall` tool shipped with Portupgrade, it will also fail (see Listing 19). This is a strength of using the packages system, not a weakness. We don't want to break the system by removing a package on which others depend. What if we decided to remove Mutt? We could check what depends on it using `pkg_info` again.

```
freebsd7# pkg_info -Rx mutt
Information for mutt-1.4.2.3_3:
```

Nothing depends on Mutt. So, if we wanted to, we could simply delete it using `pkg_deinstall`

Listing 25. Updating the portsdb used by portupgrade

```
freebsd7# portsdb -u
[Updating the portsdb <format:bdb_btree> in /usr/ports ... - 20618 port
entries found .......1000.......2000.......3000.......4000.........5000...
6000.........7000.........8000.........9000.........10000.........11000...
12000.......13000.......14000.......15000.........16000.........17000...
18000.......19000.......20000. ..... done]
```

Listing 26. Removing curl using pkg_deinstall

```
freebsd7# pkg_deinstall -R curl
--->  Deinstalling 'curl-7.19.6_1'
--->  Deinstalling 'ca_root_nss-3.11.9_2'
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 9 packages found
(-1 +0) (...) done]
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 8 packages found
(-1 +0) (...) done]
```

Listing 27. Finding the screen port in /usr/ports

```
freebsd7# cd /usr/ports
freebsd7# make search name=screen
...edited...
Port:   screen-4.0.3_6
Path:   /usr/ports/sysutils/screen
Info:   A multi-screen window manager
Maint:   cy@FreeBSD.org
B-deps:
R-deps:   gettext-0.17_1 libiconv-1.13.1 texinfo-4.11
WWW:   http://www.gnu.org/software/screen/
...truncated...
```

Listing 28. The screen port is located

```
reebsd7# cd /usr/ports/sysutils/screen
freebsd7# ls -al
total 32
drwxr-xr-x    3 root  wheel    512 Aug 24 16:46 .
drwxr-xr-x  917 root  wheel  17920 Aug 25 10:37 ..
-rw-r--r--    1 root  wheel   2366 Feb 23  2009 Makefile
-rw-r--r--    1 root  wheel    193 Oct 26  2006 distinfo
drwxr-xr-x    2 root  wheel    512 Aug 24 16:46 files
-rw-r--r--    1 root  wheel    554 Dec 27  2002 pkg-descr
-rw-r--r--    1 root  wheel    853 Aug 30  2004 pkg-plist
```

Listing 29. Running make showconfig for screen

```
freebsd7# make showconfig
===> The following configuration options are available for screen-4.0.3_6:
     CJK=OFF (default) "Treat CJK ambiguous characters as full width"
     INFO=ON (default) "Build and install info documentation"
     MAN=ON (default) "Build and install man pages"
     NETHACK=ON (default) "Enable nethack-style messages"
     XTERM_256=OFF (default) "Enable support for 256 colour xterm"
     HOSTINLOCKED=OFF (default) "Print user@host in locked message"
     SHOWENC=OFF (default) "Show encoding on the status line"
===> Use 'make config' to modify these settings
```

or `pkg_delete`. However, when we installed Mutt, it brought 8 dependencies along with it. Wouldn't it be good to remove those as well? We can use the `pkg_deinstall` command with the `-R` switch for that purpose (see Listing 20). We've now completely removed Mutt and the packages on which Mutt depended.

Identifying and Removing Unwanted Packages

For the purposes of the next example, I install Nmap and Tcpflow (see Listing 21). Let's imagine that a while passes, and later we'd like to perform some housecleaning on our installed packages.

I periodically install packages for a single task, and then leave them behind. To perform housecleaning, I prefer using the `pkg_cutleaves` tool (see Listing 22).

Next I invoke `pkg_cutleaves`. I'm looking for packages I'd like to remove. Nmap and Tcpflow catch my eye. When I want to keep a package, I hit [return] to keep it. When I want to delete a package, I hit d. When asked if I want to go on with new leaf packages, I enter y and continue the process (see Listing 23).

As we can see, the result of this process was removing Nmap, Tcpflow, and all of their dependencies. If we knew from the outset what we wanted to delete, we could have run `pkg_deinstall` as shown earlier. Here I like to use the *browsing* nature of `pkg_cutleaves` to identify packages which I don't necessarily realize I want to delete from the beginning.

Preparing to Build and Install Packages Using the Ports Tree

Throughout this article we have installed packages installed by the FreeBSD project. However, because we have the ports tree installed on our system, we can build and install our own packages.

Earlier we updated our ports tree using Portsnap. Here we will update it again (see listing 24).

After running `portsnap fetch` and `portsnap update`, we update the INDEX-7.db used by Portupgrade (see Listing 25). To keep it clear, Portsnap updates `/usr/ports/INDEX-7` and portsdb updates `/usr/ports/INDEX-7.db`.

For the following examples we will deinstall Curl and its dependencies, and then reinstall them later (see Listing 26).

For this example we will install the Screen application using the ports tree.

We'll start by using the port as an example of how to install a package. First we have to locate the port. We can use the `make search name=` command in the `/usr/ports` directory (see Listing 27).

Here we see that sysutils/screen is the port we want (see Listing 28).

These are the files we will need to build a package using this port. To determine if there are any dependencies required to build a package from this port, we can use the following command.

Listing 30. Results of running make showconfig for screen port

```
freebsd7# ls /var/db/ports
screen

freebsd7# ls /var/db/ports/screen/
options

freebsd7# cat /var/db/ports/screen/options
# This file is auto-generated by 'make config'.
# No user-servicable parts inside!
# Options for screen-4.0.3_6
_OPTIONS_READ=screen-4.0.3_6
WITHOUT_CJK=true
WITH_INFO=true
WITH_MAN=true
WITH_NETHACK=true
WITHOUT_XTERM_256=true
WITHOUT_HOSTINLOCKED=true
WITHOUT_SHOWENC=true
```

Listing 31. Running make for the screen port

```
freebsd7# make
===>  Found saved configuration for screen-4.0.3_6
=> screen-4.0.3.tar.gz doesn't seem to exist in /usr/ports/distfiles/.
=> Attempting to fetch from ftp://ftp.uni-erlangen.de/pub/utilities/screen/.
screen-4.0.3.tar.gz                           100% of  820 kB  562 kBps
===>  Extracting for screen-4.0.3_6
=> MD5 Checksum OK for screen-4.0.3.tar.gz.
=> SHA256 Checksum OK for screen-4.0.3.tar.gz.
===>  Patching for screen-4.0.3_6
===>  Applying FreeBSD patches for screen-4.0.3_6
===>  Configuring for screen-4.0.3_6
this is screen version 4.0.3
checking for gcc... cc
checking for C compiler default output... a.out
checking whether the C compiler works... yes
...edited...
cc -c -I. -I.    -O2 -fno-strict-aliasing -pipe encoding.c
cc  -o screen screen.o ansi.o fileio.o mark.o misc.o resize.o socket.o
search.o tty.o term.o window.o utmp.o loadav.o putenv.o help.o  termcap.o
input.o attacher.o pty.o process.o display.o comm.o  kmapdef.o acls.o
braille.o braille_tsi.o logfile.o layer.o  sched.o teln.o nethack.o encoding.o
-ltermcap  -lutil -lutil -lcrypt
```

```
freebsd7# make pretty-print-build-
depends-list
```

There are no dependencies to build the package. We can also see if any packages are required to run the package once installed.

```
freebsd7# make pretty-print-run-
depends-list
```

There are no dependencies to run the package. The next command I like to run when encountering a new port is `make showconfig`. This command will show the options that will be set by default when building the package from the ports tree. The default settings are used to build the package provided by the FreeBSD project (see Listing 29).

We can run `make config` to change or just view these settings. This starts a Curses window. We leave the configuration as-is but hit OK to exit. Running `make config` has created the following entries in the `/var/db/ports` directory (see Listing 31).

The ports tree will use these options when building the package.

Building and Installing Packages Using the Ports Tree: A Simple Example

At this point we are ready to proceed. In the `/usr/ports/sysutils/screen` directory, run `make` (see Listing 31).

To install we run 'make install (see Listing 32). Screen is now installed.

Building and Installing Packages Using the Ports Tree: A More Complicated Example

For a more complicated example, let's install Curl using the ports tree. To install Curl via the ports tree, we need to know where it lives. We might remember it from the ftp/curl directory at the beginning of the article, but if we aren't sure we can again use the `make search name=` command in the `/usr/ports` directory (see Listing 33).

The first option is what we want, but many other programs with `curl` in their name are listed. In addition to running `make search name=` we could also use `make search key=` to specify a keyword for searching. We see `ftp/curl` has what we want, we change there. Again we run `make showconfig` to see available options (see Listing 34). If we run *make config*, we'll see a Curses interface like the following. Here I have enabled the LIBSSH2 option, which was off by default.

The ability to modify a package to meet local requirements, but then manage that package using standard tools, is one of the great strengths of the FreeBSD ports tree.

After selecting OK, I run `make showconfig` again and notice the change (see Listing 35).

Next I like to see packages that are required to build this package.

```
freebsd7# make pretty-print-build-
depends-list
This port requires package(s) "perl-
5.8.9_3" to build.
```

Listing 32. Running make install for the screen port

```
freebsd7# make install
===>  Installing for screen-4.0.3_6
===>   Generating temporary packing list
===>  Checking if sysutils/screen already installed
...editd...
===>   Registering installation for screen-4.0.3_6
===> SECURITY REPORT:
      This port has installed the following binaries which execute with
      increased privileges.
/usr/local/bin/screen
      If there are vulnerabilities in these programs there may be a security
      risk to the system. FreeBSD makes no guarantee about the security of
      ports included in the Ports Collection. Please type 'make deinstall'
      to deinstall the port if this is a concern.
      For more information, and contact details about the security
      status of this software, see the following webpage:
http://www.gnu.org/software/screen/
freebsd7# pkgdb -vu
--->  Updating the pkgdb
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 9 packages found
(-0 +1) . done]
freebsd7# rehash
freebsd7# which screen
/usr/local/bin/screen
```

Listing 33. Finding the curl port

```
freebsd7# cd /usr/ports
freebsd7# make search name=curl
Port:   curl-7.19.6_1
Path:   /usr/ports/ftp/curl
Info:   Non-interactive tool to get files from FTP, GOPHER, HTTP(S) servers
Maint:   roam@FreeBSD.org
B-deps:   perl-5.8.9_3
R-deps:   ca_root_nss-3.11.9_2
WWW:   http://curl.haxx.se/
Port:   curlpp-0.7.0_1
Path:   /usr/ports/ftp/curlpp
Info:   A C++ wrapper for libcurl
Maint:   roam@FreeBSD.org
B-deps:   ca_root_nss-3.11.9_2 curl-7.19.6_1
R-deps:   ca_root_nss-3.11.9_2 curl-7.19.6_1
WWW:
...truncated...
```

We already have Perl installed, so we don't have to worry about it. If Perl were not installed, we might consider installing the Perl package ourselves. If we did not install the Perl package, using the ports tree would result in building Perl and its dependencies (if any) from the ports tree as well. Now I see what packages are required to run this package, once installed.

```
freebsd7# make pretty-print-run-
depends-list
This port requires package(s) "ca_root_
nss-3.11.9_2" to run.
```

That makes sense. We already saw that when Curl was installed, the `ca_root_nss` package was listed as a dependency.

However, in the next section we will find that this output is not complete due to the customization for `libssh2` that we introduced.

To simply install Curl, we could use *make* again. However, we saw that `ca_root_nss-3.11.9_2` is a runtime dependency. We can install the package manually first before installing Curl via the ports tree (see Listing 36). Now, when we install Curl via the ports tree, we don't have to worry about the dependency being installed through the ports tree (see Listing 37). Now we run *make install* (see Listing 38). During installation, `libssh2` was found to be a dependency, based on the customization we made. We can see the dependency using `pkg_tree`.

```
freebsd7# pkg_tree curl
curl-7.19.6_1
|\__ libssh2-1.2,2
 \__ ca_root_nss-3.11.9_2
```

If we want to create a package for Curl, we can use the `make package` command (see Listing 39). If we want to make the package and its dependencies, we use '`make package-recursive`' (see Listing 40). Note that using the `make package-recursive` command means you don't have to run `make install`. With FreeBSD, there is not a way to make a package but avoid installation.

```
freebsd7# ls /usr/ports/packages/All
ca_root_nss-3.11.9_2.tbz   perl-5.8.9_3.tbz
curl-7.19.6_1.tbz        portaudit-
0.5.13.tbz
libssh2-1.2,2.tbz
```

Install Packages Built on One System to Another System

Once packages are built using the ports tree, you can install them on similar systems elsewhere. For example, we can copy packages from freebsd7 to another system and install them locally. In the following example we begin on host freebsd7S and will install packages built earlier in this article (see Listing 41).

So, the new package is installed, but what if we wanted to add a package with

Listing 34. Running make showconfig for the curl port

```
freebsd7# make showconfig
===> The following configuration options are available for curl-7.19.6_1:
     CARES=off (default) "Asynchronous DNS resolution via c-ares"
     CURL_DEBUG=off (default) "Enable curl diagnostic output"
     GNUTLS=off (default) "Use GNU TLS if OPENSSL is OFF"
     IPV6=on (default) "IPv6 support"
     KERBEROS4=off (default) "Kerberos 4 authentication"
     LDAP=off (default) "LDAP support"
     LDAPS=off (default) "LDAPS support (requires LDAP and SSL)"
     LIBIDN=off (default) "Internationalized Domain Names via libidn"
     LIBSSH2=off (default) "SCP/SFTP support via libssh2"
     NTLM=off (default) "NTLM authentication"
     OPENSSL=on (default) "OpenSSL support"
     PROXY=on (default) "Proxy support"
     TRACKMEMORY=off (default) "Enable curl memory diagnostic output"
===> Use 'make config' to modify these settings
```

Listing 35. Changes after running make showconfig

```
freebsd7# make showconfig
===> The following configuration options are available for curl-7.19.6_1:
     CARES=off "Asynchronous DNS resolution via c-ares"
     CURL_DEBUG=off "Enable curl diagnostic output"
     GNUTLS=off "Use GNU TLS if OPENSSL is OFF"
     IPV6=on "IPv6 support"
     KERBEROS4=off "Kerberos 4 authentication"
     LDAP=off "LDAP support"
     LDAPS=off "LDAPS support (requires LDAP and SSL)"
     LIBIDN=off "Internationalized Domain Names via libidn"
     LIBSSH2=on "SCP/SFTP support via libssh2"
     NTLM=off "NTLM authentication"
     OPENSSL=on "OpenSSL support"
     PROXY=on "Proxy support"
     TRACKMEMORY=off "Enable curl memory diagnostic output"
===> Use 'make config' to modify these settings
```

Listing 36. Installing the curl package

```
freebsd7# pkg_add -r ca_root_nss
Fetching ftp://ftp.freebsd.org//pub/FreeBSD/ports/i386/packages-7-stable/
Latest/ca_root_nss.tbz... Done.
freebsd7# pkgdb -vu
--->  Updating the pkgdb
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 10 packages found
(-0 +1) . done]
```

Listing 37. Running make for the curl port

```
freebsd7# make
===>  Found saved configuration for curl-7.19.6_1
=> curl-7.19.6.tar.bz2 doesn't seem to exist in /usr/
ports/distfiles/.
=> Attempting to fetch from http://curl.haxx.se/
download/.
fetch: transfer timed out
=> Attempting to fetch from ftp://ftp.sunet.se/pub/www/
utilities/curl/.
curl-7.19.6.tar.bz2             100% of 2292 kB 2336 kBps
===>  Extracting for curl-7.19.6_1
=> MD5 Checksum OK for curl-7.19.6.tar.bz2.
=> SHA256 Checksum OK for curl-7.19.6.tar.bz2.
===>   curl-7.19.6_1 depends on file: /usr/local/bin/
perl5.8.9 - found
===>  Patching for curl-7.19.6_1
===>   curl-7.19.6_1 depends on file: /usr/local/bin/
perl5.8.9 - found
===>  Applying FreeBSD patches for curl-7.19.6_1
===>   curl-7.19.6_1 depends on file: /usr/local/bin/
perl5.8.9 - found
===>   curl-7.19.6_1 depends on shared library: ssh2.1
- not found
===>    Verifying install for ssh2.1 in /usr/ports/
security/libssh2
=> libssh2-1.2.tar.gz doesn't seem to exist in /usr/
ports/distfiles/.
=> Attempting to fetch from http://www.libssh2.org/
download/.
fetch: transfer timed out
=> Attempting to fetch from http://redundancy.redundancy
.org/mirror/.
fetch: http://redundancy.redundancy.org/mirror/libssh2-
1.2.tar.gz: Not Found
=> Attempting to fetch from ftp://ftp.FreeBSD.org/pub/
FreeBSD/ports/distfiles/.
libssh2-1.2.tar                 100% of  519 kB 2150 kBps
===>  Extracting for libssh2-1.2,2
=> MD5 Checksum OK for libssh2-1.2.tar.gz.
=> SHA256 Checksum OK for libssh2-1.2.tar.gz.
===>  Patching for libssh2-1.2,2
===>  Configuring for libssh2-1.2,2
checking whether to enable maintainer-specific portions
of Makefiles... no
checking for sed... /usr/bin/sed
checking for a BSD-compatible install... /usr/bin/
install -c -o root -g wheel
checking whether build environment is sane... yes
...edited...
===>   Registering installation for libssh2-1.2,2
===>   Returning to build of curl-7.19.6_1
===>  Configuring for curl-7.19.6_1
checking whether to enable maintainer-specific portions
of Makefiles... no
checking whether to enable debug build options... no
checking whether to enable compiler optimizer... not
specified (assuming yes)
checking whether to enable strict compiler warnings... no
checking whether to enable curl debug memory tracking... no
checking for sed... /usr/bin/sed
checking for grep... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for ar... /usr/bin/ar
checking for a BSD-compatible install... /usr/bin/
install -c -o root -g wheel
checking whether build environment is sane... yes
...edited...
Making all in examples
Making all in libcurl
```

Listing 38. Running make install for the curl port

```
freebsd7# make install
===>  Installing for curl-7.19.6_1
===>   curl-7.19.6_1 depends on file: /usr/local/share/
certs/ca-root-nss.crt - found
===>   curl-7.19.6_1 depends on shared library: ssh2.1 - found
===>   Generating temporary packing list
===>  Checking if ftp/curl already installed
Making install in lib
...edited...
===>   Registering installation for curl-7.19.6_1
===> SECURITY REPORT:
      This port has installed the following files which
may act as network
      servers and may therefore pose a remote security
risk to the system.
/usr/local/lib/libcurl.so.5
      If there are vulnerabilities in these programs
there may be a security
      risk to the system. FreeBSD makes no guarantee
about the security of
      ports included in the Ports Collection. Please
type 'make deinstall'
      to deinstall the port if this is a concern.
      For more information, and contact details about
the security
      status of this software, see the following
webpage:
http://curl.haxx.se/
freebsd7# pkgdb -vu
--->  Updating the pkgdb
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg
... - 12 packages found (-0 +2) .. done]
```

dependencies? For examples like this, we could mount the remote system's /usr/ports/packages directory using NFS, and add from there. The remote system here is freebsd7, or 172.16.134.128. I recommend making a read-only mount (via -o ro) so that the NFS client does not accidentally alter the server (see Listing 42).

Listing 39. Running make package for the curl port

```
freebsd7# make package
===>  Building package for curl-7.19.6_1
Creating package /usr/ports/packages/All/curl-7.19.6_1.tbz

Registering depends: ca_root_nss-3.11.9_2 libssh2-1.2,2.
Creating bzip'd tar ball in '/usr/ports/packages/All/curl-7.19.6_1.tbz'
```

Listing 40. Running make package-recursive for the curl port

```
freebsd7# make package-recursive
===>   Generating temporary packing list
Creating package /usr/ports/packages/All/perl-5.8.9_3.tbz
Registering depends:.
Registering conflicts: perl-5.6.* perl-5.10.* perl-threaded-5.10.*.
Creating bzip'd tar ball in '/usr/ports/packages/All/perl-5.8.9_3.tbz'
Creating package /usr/ports/packages/All/libssh2-1.2,2.tbz
Registering depends:.
Creating bzip'd tar ball in '/usr/ports/packages/All/libssh2-1.2,2.tbz'
rmdir: /usr/ports/security/libssh2/work: Directory not empty
*** Error code 1 (ignored)
===>   Generating temporary packing list
Creating package /usr/ports/packages/All/ca_root_nss-3.11.9_2.tbz
Registering depends:.
Creating bzip'd tar ball in '/usr/ports/packages/All/ca_root_nss-3.11.9_2.tbz'
```

Listing 41. Copying and installing a package from another system

```
freebsd7S# uname -a
FreeBSD freebsd7S.taosecurity.com 7.2-STABLE FreeBSD 7.2-STABLE #2: Sat Aug
22 17:12:42 EDT 2009     root@freebsd7.localdomain:/usr/obj/usr/src/sys/
FREEBSD7  i386
freebsd7S# mkdir -p /usr/ports/packages/All
freebsd7S# sftp analyst@172.16.134.128
Connecting to 172.16.134.128...
Password:
sftp> cd /usr/ports/packages/All
sftp> ls
ca_root_nss-3.11.9_2.tbz  curl-7.19.6_1.tbz         libssh2-1.2,2.tbz
perl-5.8.9_3.tbz          portaudit-0.5.13.tbz
sftp> get ca_root_nss-3.11.9_2.tbz
Fetching /usr/ports/packages/All/ca_root_nss-3.11.9_2.tbz to ca_root_nss-
3.11.9_2.tbz
/usr/ports/packages/All/ca_root_nss-3.11.9_2. 100%  169KB 168.7KB/s   00:00
sftp> quit
freebsd7S# pkg_add ca_root_nss-3.11.9_2.tbz
freebsd7S# pkg_info | grep ca_root
ca_root_nss-3.11.9_2 The root certificate bundle from the Mozilla Project
```

Curl and its dependencies have now been installed over NFS from the freebsd7 system. This example demonstrates how a centralized system (in this case, freebsd7) could serve as a local site ports tree and package repository, and client systems (like freebsd7S) could install packages from the local repository. In fact, the clients would not have to maintain their own ports trees.

Let's show how mounting /usr/ports from the package repository freebsd7 helps the client freebsdS learn what packages need updating. First, for this particular installation, we know that our clients will need the sysutils/cmdwatch utility, so we make a package of it on our package builder freebsd7 (see Listing 43). Now we turn to the package client, freebsd7S, and mount the package builder's /usr/ports directory (see Listing 44). We can run pkgdb -vu on freebsd7S because it stores the package database used by Portupgrade in /var/db/pkg on the local system.

```
freebsd7S# pkgdb -vu
--->  Updating the pkgdb
[Rebuilding the pkgdb <format:bdb_
btree> in /var/db/pkg ... - 12 packages
found (-0 +12) ........... done]
```

Now we run portversion -v to see which packages need updating on the client. The client uses NFS to compare to the versions on the package builder (see Listing 45).

We see that cmdwatch and screen need updating. On the client we invoke Portupgrade using the -vaPP switches (see Listing 46).

We see that Portupgrade did not find a screen package on the package builder (in /usr/ports/packages/All), and when it failed it tried to find a package on a remote FreeBSD server. That failed too, because the FreeBSD project does not build screen packages. The project recommends users build their own packages for screen. However, the newest version of cmdwatch was installed, using the package /usr/ports/packages/All/cmdwatch-0.2.0_2.tbz.

Installing Screen Using a Remote FreeBSD Ports Tree

What do we do about screen? It turns out that we can work around this problem.

Listing 42. Installing the curl package over NFS

```
freebsd7S# mount -t nfs -o ro 172.16.134.128:/usr/ports/
packages /usr/ports/packages
freebsd7S# mount
/dev/ad0s1a on / (ufs, local)
devfs on /dev (devfs, local)
/dev/ad0s1f on /home (ufs, local, soft-updates)
/dev/ad0s1g on /tmp (ufs, local, soft-updates)
/dev/ad0s1d on /usr (ufs, local, soft-updates)
/dev/ad0s1e on /var (ufs, local, soft-updates)
172.16.134.128:/usr/ports/packages on /usr/ports/
packages (nfs, read-only)
freebsd7S# cd /usr/ports/packages/All
freebsd7S# ls
ca_root_nss-3.11.9_2.tbz   perl-5.8.9_3.tbz
curl-7.19.6_1.tbz      portaudit-0.5.13.tbz
libssh2-1.2,2.tbz
freebsd7S# pkg_add -v curl-7.19.6_1.tbz
Requested space: 4525K bytes, free space: 455M bytes in
/var/tmp/instmp.0YHgDV
Package 'curl-7.19.6_1' depends on 'libssh2-1.2,2' with
'security/libssh2' origin.
Loading it from /usr/ports/packages/All/libssh2-
1.2,2.tbz.
Requested space: 711K bytes, free space: 452M bytes in
/var/tmp/instmp.6EzFeD
extract: Package name is libssh2-1.2,2
extract: CWD to /usr/local
extract: /usr/local/include/libssh2.h
...edited...
extract: /usr/local/man/man3/libssh2_version.3.gz
extract: execute '/sbin/ldconfig -m /usr/local/lib'
extract: CWD to .
Running mtree for libssh2-1.2,2..
mtree -U -f +MTREE_DIRS -d -e -p /usr/local >/dev/null
Attempting to record package into /var/db/pkg/libssh2-
1.2,2..
Package libssh2-1.2,2 registered in /var/db/pkg/libssh2-
1.2,2
Package 'curl-7.19.6_1' depends on 'ca_root_nss-3.11.9_
2' with 'security/ca_root_nss' origin.
 - already installed.
extract: Package name is curl-7.19.6_1
extract: CWD to /usr/local
extract: /usr/local/man/man1/curl.1.gz
...edited...
extract: /usr/local/share/examples/curl/threaded-ssl.c
extract: execute '/sbin/ldconfig -m /usr/local/lib'
extract: CWD to .
Running mtree for curl-7.19.6_1..
mtree -U -f +MTREE_DIRS -d -e -p /usr/local >/dev/null
Attempting to record package into /var/db/pkg/curl-
7.19.6_1..
Trying to record dependency on package 'libssh2-1.2,2'
with 'security/libssh2' origin.
Trying to record dependency on package 'ca_root_nss-
3.11.9_2' with 'security/ca_root_nss' origin.
Package curl-7.19.6_1 registered in /var/db/pkg/curl-
7.19.6_1
freebsd7S# cd
freebsd7S# umount /usr/ports/packages
```

Listing 43. Creating the cmdwatch package

```
freebsd7# cd /usr/ports/sysutils/cmdwatch
freebsd7# make package
=> cmdwatch-0.2.0.tar.gz doesn't seem to exist in /usr/
ports/distfiles/.
=> Attempting to fetch from http://www.chruetertee.ch/
files/download/.
cmdwatch-0.2.0.tar.gz          100% of   11 kB   66 kBps
===>  Extracting for cmdwatch-0.2.0_2
=> MD5 Checksum OK for cmdwatch-0.2.0.tar.gz.
=> SHA256 Checksum OK for cmdwatch-0.2.0.tar.gz.
===>  Patching for cmdwatch-0.2.0_2
===>  Applying FreeBSD patches for cmdwatch-0.2.0_2
===>  Configuring for cmdwatch-0.2.0_2
===>  Building for cmdwatch-0.2.0_2
Making cmdwatch...getopt.c: In function '_getopt_
internal':
...edited...
 done.
===>  Installing for cmdwatch-0.2.0_2
===>   Generating temporary packing list
===>  Checking if sysutils/cmdwatch already installed
Making cmdwatch... done.
Installing cmdwatch
===>   Compressing manual pages for cmdwatch-0.2.0_2
===>   Registering installation for cmdwatch-0.2.0_2
===>  Building package for cmdwatch-0.2.0_2
Creating package /usr/ports/packages/All/cmdwatch-0.2.0_
2.tbz
Registering depends:.
Creating bzip'd tar ball in '/usr/ports/packages/All/
cmdwatch-0.2.0_2.tbz'
freebsd7# pkgdb -vu
--->  Updating the pkgdb
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg
... - 13 packages found (-0 +1) . done]
```

Listing 44. Mounting the package builder using NFS

```
freebsd7S# mount -t nfs -o ro 172.16.134.128:/usr/ports
/usr/ports
freebsd7S# mount
/dev/ad0s1a on / (ufs, local)
devfs on /dev (devfs, local)
/dev/ad0s1f on /home (ufs, local, soft-updates)
/dev/ad0s1g on /tmp (ufs, local, soft-updates)
/dev/ad0s1d on /usr (ufs, local, soft-updates)
/dev/ad0s1e on /var (ufs, local, soft-updates)
172.16.134.128:/usr/ports on /usr/ports (nfs, read-only)
```

Listing 45. Running portversion

```
freebsd7S# portversion -v
ca_root_nss-3.11.9_2          =  up-to-date with port
cmdwatch-0.2.0_1            <  needs updating (port has 0.2.0_2)
curl-7.19.6_1                 =  up-to-date with port
cvsup-without-gui-16.1h_4     =  up-to-date with port
db41-4.1.25_4                 =  up-to-date with port
libssh2-1.2,2                 =  up-to-date with port
perl-5.8.9_3                  =  up-to-date with port
pkg_cutleaves-20090810        =  up-to-date with port
portupgrade-2.4.6_3,2         =  up-to-date with port
ruby-1.8.7.160_4,1            =  up-to-date with port
ruby18-bdb-0.6.5_1            =  up-to-date with port
screen-4.0.3_5            <  needs updating (port has 4.0.3_6)
```

Listing 46. Running portupgrade

```
freebsd7S# portupgrade -vaPP
--->  Session started at: Tue, 25 Aug 2009 14:56:56 -0400
--->  Checking for the latest package of 'sysutils/screen'
--->  Fetching the package(s) for 'screen-4.0.3_6'
(sysutils/screen)
--->  Fetching screen-4.0.3_6
++ Will try the following sites in the order named:
   ftp://ftp.FreeBSD.org//pub/FreeBSD/ports/i386/
packages-7-stable/
--->  Invoking a command: /usr/bin/fetch -o '/var/
tmp/portupgrade69eLJ2VS/screen-4.0.3_6.tbz' 'ftp://
ftp.FreeBSD.org/pub/FreeBSD/ports/i386/packages-7-stable/
All/screen-4.0.3_6.tbz'
fetch: ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/
i386/packages-7-stable/All/screen-4.0.3_6.tbz: File
unavailable (e.g., file not found, no access)
** The command returned a non-zero exit status: 1
** Failed to fetch ftp://ftp.FreeBSD.org/pub/FreeBSD/
ports/i386/packages-7-stable/All/screen-4.0.3_6.tbz
--->  Invoking a command: /usr/bin/fetch -o '/var/
tmp/portupgrade69eLJ2VS/screen-4.0.3_6.tgz' 'ftp://
ftp.FreeBSD.org/pub/FreeBSD/ports/i386/packages-7-stable/
All/screen-4.0.3_6.tgz'
fetch: ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/
i386/packages-7-stable/All/screen-4.0.3_6.tgz: File
unavailable (e.g., file not found, no access)
** The command returned a non-zero exit status: 1
** Failed to fetch ftp://ftp.FreeBSD.org/pub/FreeBSD/
ports/i386/packages-7-stable/All/screen-4.0.3_6.tgz
** Failed to fetch screen-4.0.3_6
--->  Listing the results (+:done / -:ignored / *:skipped / !:failed)
   ! screen-4.0.3_6   (fetch error)
--->  Packages processed: 0 done, 0 ignored, 0 skipped and
1 failed
--->  Fetching the latest package(s) for 'screen' (sysutils/screen)
--->  Fetching screen
++ Will try the following sites in the order named:
   ftp://ftp.FreeBSD.org//pub/FreeBSD/ports/i386/
packages-7-stable/
--->  Invoking a command: /usr/bin/fetch -o '/var/tmp/
portupgradeKmGTSv48/screen.tbz' 'ftp://ftp.FreeBSD.org/pub/
FreeBSD/ports/i386/packages-7-stable/Latest/screen.tbz'
fetch: ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/i386/
packages-7-stable/Latest/screen.tbz: File unavailable
(e.g., file not found, no access)
** The command returned a non-zero exit status: 1
** Failed to fetch ftp://ftp.FreeBSD.org/pub/FreeBSD/
ports/i386/packages-7-stable/Latest/screen.tbz
--->  Invoking a command: /usr/bin/fetch -o '/var/tmp/
portupgradeKmGTSv48/screen.tgz' 'ftp://ftp.FreeBSD.org/pub/
FreeBSD/ports/i386/packages-7-stable/Latest/screen.tgz'
fetch: ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/i386/
packages-7-stable/Latest/screen.tgz: File unavailable
(e.g., file not found, no access)
** The command returned a non-zero exit status: 1
** Failed to fetch ftp://ftp.FreeBSD.org/pub/FreeBSD/
ports/i386/packages-7-stable/Latest/screen.tgz
** Failed to fetch screen
--->  Listing the results (+:done / -:ignored / *:skipped
/ !:failed)
   ! screen@   (fetch error)
--->  Packages processed: 0 done, 0 ignored, 0 skipped
and 1 failed
** Could not find the latest version (4.0.3_6)
** No package available: sysutils/screen
--->  ** Upgrade tasks 2: 0 done, 0 ignored, 0 skipped
and 1 failed
--->  Checking for the latest package of 'sysutils/cmdwatch'
--->  Found a package of 'sysutils/cmdwatch': /usr/ports/
packages/All/cmdwatch-0.2.0_2.tbz (cmdwatch-0.2.0_2)
--->  Upgrade of sysutils/cmdwatch started at: Tue, 25
Aug 2009 14:58:01 -0400
--->  Upgrading 'cmdwatch-0.2.0_1' to 'cmdwatch-0.2.0_2'
(sysutils/cmdwatch) using a package
--->  Updating dependency info
--->  Uninstallation of cmdwatch-0.2.0_1 started at: Tue, 25 Aug
2009 14:58:02 -0400
```

Listing 46. Running portupgrade (cont)

```
--->  Fixing up dependencies before creating a package
--->  Backing up the old version
--->  Uninstalling the old version
--->  Deinstalling 'cmdwatch-0.2.0_1'
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 11 packages found
(-1 +0)  (...) done]
--->  Uninstallation of cmdwatch-0.2.0_1 ended at: Tue, 25 Aug 2009 14:58:28
-0400 (consumed 00:00:25)
--->  Installation of cmdwatch-0.2.0_2 started at: Tue, 25 Aug 2009 14:58:28 -0400
--->  Installing the new version via the package
--->  Removing temporary files and directories
--->  Removing old package'
--->  Installation of cmdwatch-0.2.0_2 ended at: Tue, 25 Aug 2009 14:58:31 -
0400 (consumed 00:00:02)
--->  Cleaning out obsolete shared libraries
[Updating the pkgdb <format:bdb_btree> in /var/db/pkg ... - 12
packages found (-0 +1) . done]
--->  Upgrade of sysutils/cmdwatch ended at: Tue, 25 Aug 2009 14:58:53 -0400
(consumed 00:00:51)
--->  ** Upgrade tasks 2: 1 done, 0 ignored, 0 skipped and 1 failed
--->  Listing the results (+:done / -:ignored / *:skipped / !:failed)
   ! sysutils/screen (screen-4.0.3_5)   (package not found)
   + sysutils/cmdwatch (cmdwatch-0.2.0_1)
--->  Packages processed: 1 done, 0 ignored, 0 skipped and 1 failed
--->  Session ended at: Tue, 25 Aug 2009 14:59:12 -0400 (consumed 00:02:16)
freebsd7S# portversion -v
ca_root_nss-3.11.9_2        =  up-to-date with port
cmdwatch-0.2.0_2            =  up-to-date with port
curl-7.19.6_1               =  up-to-date with port
cvsup-without-gui-16.1h_4   =  up-to-date with port
db41-4.1.25_4               =  up-to-date with port
libssh2-1.2,2               =  up-to-date with port
perl-5.8.9_3                =  up-to-date with port
pkg_cutleaves-20090810      =  up-to-date with port
portupgrade-2.4.6_3,2       =  up-to-date with port
ruby-1.8.7.160_4,1          =  up-to-date with port
ruby18-bdb-0.6.5_1          =  up-to-date with port
screen-4.0.3_5              <  needs updating (port has 4.0.3_6)
```

Listing 47. Common package upgrade process

```
1. setenv HTTP_PROXY [insert your proxy]
2. setenv PACKAGESITE ftp://ftp[X].freebsd.org//pub/FreeBSD/ports/i386/packages-7-
stable/Latest/ where [X] is the number of a FreeBSD FTP server near you.
3. portsnap fetch
4. portsnap update
5. portsdb -u
6. pkgdb -vu
7. portversion -v -l "<"
8. Read /usr/ports/UPDATING to see if any special instructions apply to
packages of interest.
9. portupgrade -vaPP
10. portversion -v -l "<"
```

About the Author

Richard Bejtlich is Director of Incident Response for General Electric, and serves as Principal Technologist for GE's Global Infrastructure Services division. He also writes for his blog (taosecurity.blogspot.com) and TechTarget.com, and teaches for Black Hat.

First, remove the old package using any of the methods demonstrated in this article. Next, mount the package builder's ports directory.

```
freebsd7S# mount -t nfs -o ro
172.16.134.128:/usr/ports /usr/ports
```

If `/usr/ports` doesn't exist on the client, create it. Now cd to `/usr/ports/sysutils/screen` and run the following.

```
freebsd7S#  cd /usr/ports/sysutils/
screen
freebsd7S# make WRKDIRPREFIX=/tmp
freebsd7S# make install WRKDIRPREFIX=/
tmp
```

When done, the new version of screen will be built, using the remote package builder's ports tree but by installing source code on the local system.

My Common Package Update Process

So what is the end result of this process? For individual systems, I recommend the following process. This assumes Portupgrade is installed, and that I rely on packages produced by the FreeBSD project. I also assume that Portaudit is running automatically every day already (see Listing 47).

Conclusion

I hope this article has helped you understand the different ways to keep FreeBSD applications up-to-date. It is by no means comprehensive, but by following it you hopefully can judge the different ways to keep your applications current.

Spam control
with a stock OpenBSD install

Girish Venkatachalam

Ever since e-mails became ubiquitous unwanted e-mails or spam also known as UCE(Unsolicited Commercial E-mail) or UBE(Unsolicited Bulk e-mail) also became popular.

E-mails today form the backbone of any company no matter if it makes space crafts or leather boots. E-mail is not just a medium for communication but the most important corporate tool.

Due to many historic accidents and a combination of other factors spammers have been finding it really easy to fake e-mail sending. They send unsolicited e-mails with so much impunity and coordination that nearly every spam protection mechanism be it technical, political or regulatory has been smashed to smithereens.

Spammers are smart people and they know what they are doing. And their level of motivation is quite high because there is money in the business. People do fall for lures to get bigger private organs or easy money. Lottery wins, Nigerian widows and `p0rn` may not interest you and me; they actually annoy us but Internet still has many naive people who these spammers can bait and walk off with their money.

Spamming is a volume game. They don't care for particular users receiving their messages. They really don't care if intelligent users delete their mails or use a spam filter. As long as their profits are higher than their overheads their business model works. As I said, they are in the game for the money.

That being the case any technology solution to fight spammers has to hit them where it hurts them the most. We have to target their business model. This is easily done. Let me explain.

Spammers hardly spend any `money/resources` to send out spam. Spammers send bulk commercial e-mail using a huge network of coordinated computers working in tandem to send out millions of mails. They use open relays and they even create bogus BGP routes to send out spam. Unallocated BGP networks known as bogons come and go. Spammers come, they send out the mails and then they vanish. They operate from some other country to avoid detection. Any form of regulation is not going to fight them. We have to use technology. And OpenBSD has an excellent method to fight such brain damaged individuals. This article is about that.

How OpenBSD fights spam?

You may be wondering how an open source free operating system can be equipped with a spam filter. And you may also wonder at its effectiveness. It turns out that OpenBSD's spam filtering arsenal is the most powerful spam filtering technique on the planet. It is way too powerful compared to anything you already know like Spamassassin or any other commercial product. It wins hands down in this particular game.

And most of all it is completely free. All you need is download the latest OpenBSD ISO, install it on a PC and set things up. You have to run this machine in front of your e-mail server machine since OpenBSD `spamd(8)` does not even allow spam to come in. It saves your bandwidth and e-mail `storage/archival` costs.

Moreover this does not require any manual intervention or maintenance. No babysitting necessary like Spamassassin and certainly no false positives problem in which you lose legitimate e-mails.

It is the ultimate spam filter!

But what it does is not really spam filtering. It performs spam control by not even letting spam in. Spammers get an error message and they cannot survive our tests since that costs them resources and money. And their business model does not allow that.

OpenBSD does tarpitting or teergrubing in such a fashion that legitimate mail senders don't feel anything but it hits the spammer. And it hits him real hard. Ultimately you as a user win without losing anything.

And you end up losing a lot when you use a content scanning spam filter like Spamassassin. Spamassassin slows your e-mails since it does content scanning and you lose important e-mail. It also requires heavy maintenance and it wastes mail server storage space and your network bandwidth. It deletes spam mails after receiving them and people have to manually pass false positives or get rid of real spam.

Whereas OpenBSD works in a completely different manner. It relies on traffic shaping at the TCP layer to achieve spam control.

Here is a diagrammatic representation of how OpenBSD does spam filtering see Figure 1.

There are three UNIX daemons of interest here. `spamd(8)` is the master daemon that runs a fake SMTP daemon at port 8025. Run this command as root on an OpenBSD machine.

```
# /usr/libexec/spamd
```

After that try this command.

```
$ nc -v localhost 8025
```

You will find how OpenBSD spamd implements the tarpit mechanism to hurt spammers.

Here onwards we are going to get really technical. Please be warned that this section is meant for people who are very familiar with the OpenBSD OS. People with a thorough grounding on OpenBSD's internals and firewall software `pf(4)` will benefit the most from the rest of the article. However you can read through and pick up the gaps in your understanding once you gain more familiarity with this fantastic operating system.

`spamlogd(8)` is a `libpcap(3)` infinite loop that reads the pflog0 virtual network interface to check for connections to the mail server. This daemon is important since without this, legitimate mails will not get through.

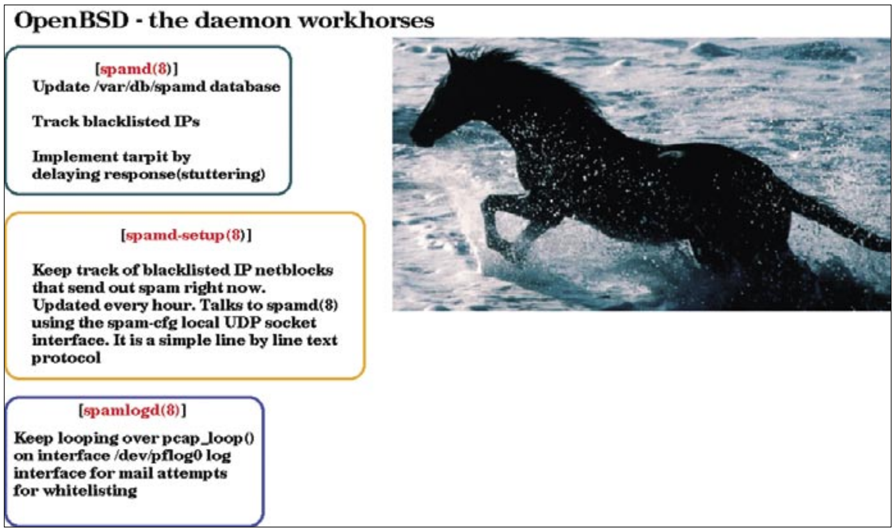
And the `spamd-setup(8)` daemon checks for the worldwide blacklists of spam senders. This daemon talks to the `spamd(8)` daemon using a simple line by line text protocol on local spamd-cfg UDP port.

You need this `/etc/pf.conf` file since it is OpenBSD's excellent firewall pf(4) that does all the networking magic for us. `pf(4)` tables are a very powerful concept for blacklisting IP addresses that misbehave in many ways. Spam control is not very different from other forms of misbehavior like launching ssh bruteforce attacks or denial of service attacks on us.

`pf(4)` tables allow us to add hosts and IP addresses dynamically based on matching rules and you can check against this list for future packets from those hosts and act in a different manner.

Let me illustrate the above concept with an example.

Let us say you have ssh bruteforce attacks coming from an IP address 1.3.4.45. You can identify this in a `pf(4)`



[spamd(8)]
Update /var/db/spamd database
Track blacklisted IPs
Implement tarpit by delaying response(stuttering)

[spamd-setup(8)]
Keep track of blacklisted IP netblocks that send out spam right now. Updated every hour. Talks to spamd(8) using the spam-cfg local UDP socket interface. It is a simple line by line text protocol

[spamlogd(8)]
Keep looping over pcap_loop() on interface /dev/pflog0 log interface for mail attempts for whitelisting

Figure 1. OpenBSF-spam-daemons

rule and you can add this IP to the table called `<badhosts>` with this rule.

```
pass inet proto tcp from any to port
$tcp_services \
    (max-src-conn 100, max-src-conn-rate
15/5, \
          overload <bathosts> flush
global)
```

And now all you have to do is plonk this line:

```
block quick from <badhosts>
```

towards the beginning of your `pf(4)` rules. You also need to declare this table beforehand.

Now getting back to spam control. This is the firewall rule file `/etc/pf.conf` that you need for spam control (see Listing 1).

For those of you who know `pf(4)` already, this file should not be cryptic at all. But for others I shall do a bit of explaining. Most lines are very clear and self explanatory. I shall only touch upon the important ones. The line:

```
rdr pass log on $ext_if proto tcp from
<white> to ($ext_if) port smtp \
   -> <mailserver> round-robin
```

does a destination NAT or TCP connection forwarding to the `<mailserver>` table declared above. The keyword `round-robin` states that if there are n hosts in the `<mailserver>` table like this:

```
table <mailserver> const { 1.2.3.5 ,
1.2.3.6, 1.2.3.7}
```

then, the rdr rule will redirect the first SMTP connection to 1.2.3.5, the second to 1.2.3.6 and the 4th back to 1.2.3.5. The other line of interest is this one:

```
pass in log on $ext_if inet proto tcp
to <mailserver> port smtp
```

This line logs the SMTP connections the real mailserver that could be running MS Exchange or sendmail or Postfix or whatever.

This is done because without the `pflog0` virtual interface seeing the successful SMTP connections, real mails will never get through as the well behaved IP addresses will not get a chance to get noticed by spamd.

This job is done by `spamlogd(8)` deamon in a `pcap_capture_live()` loop. The actual whitelisting is performed by `spamd(8)` but `spamlogd(8)` notices the packets on the log interface. I know my English is bit cryptic but I hope you get the idea.

And you will also know in a minute that this requires this sysctl setting.

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

You will also notice that this firewall config ensures that mails are forwarded to 3 mailservers 1.2.3.5, 1.2.3.6 and 1.2.3.7 one after another in a round robin fashion.

This is useful when you have three MX records like this (see Listing 2).

The other thing you could do is failover between two OpenBSD boxes by doing CARP failover between them.

The thing about CARP is that it is a patent unencumbered version of the Cisco VRRP protocol(*virtual router redundancy protocol*) and it works with any service you wish to offer with a 100% uptime guarantee. CARP relies on the IP protocol 112. (Do a grep of CARP with `/etc/protocols`)

CARP works remarkably well in an environment where you have colocated web servers or mail servers or just about any service you offer on top of IP. It could even be a UDP service. CARP is utterly painless and its real simplicity sometimes can be quite dumbfounding since it does a lot of work behind the scenes.

This does not get you TCP connection handover but this is pretty close to what you can get with minimal investment on hardware and software.

Just run the commands on machine A and machine B. On machine A, assume the network interface is `vic0`.

```
# ifconfig carp0 192.168.1.100 carpdev
vic0 vhid 1
```

On machine B, assume network interface is `fxp0`.

```
#ifconfig carp0 192.168.1.100 carpdev
fxp0 vhid 1
```

Now all you have to do is use the virtual IP 192.168.1.100 of the carp virtual interface and you have a load balanced spam filter that can give you 100% uptime and it does load sharing with 3 mail servers and it also does not allow spam to enter your network thus saving you bandwidth.

The most interesting thing above all that is that it is 100% free of cost. The source code is free, you can do whatever you want with it and still I wonder why nobody is using this method to fight spam instead of Spamassassin.

Do you know?

I have an idea why the world has not yet woken up to this technology. People do not know the value of open source and this has got to do with the thinking that anything available for free of cost and with no strings attached must be automatically bad.

Unfortunately we live in a world where open source dominates technological innovation especially in the UNIX world.

OpenBSD attracts very smart minds to its folder and its transparent developer culture and no nonsense attitude has consistently brought about some of the best technologies in firewalling, advanced networking techniques and of course e-mail spam control.

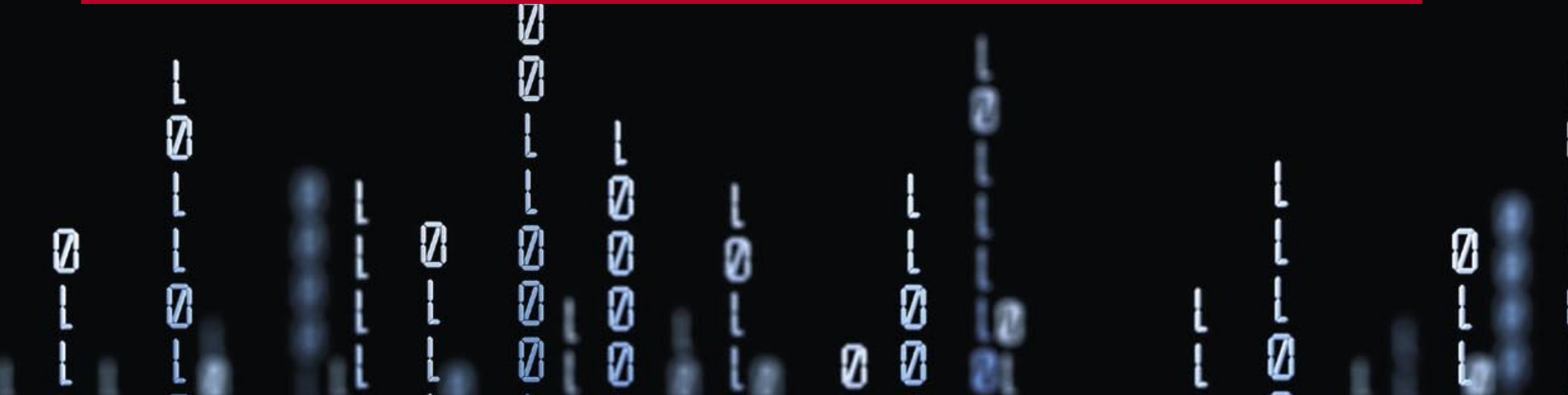
Crypto is just in passing. OpenBSD has been having the best IPSec suite for many years now. And there are many other facilities too. But security is built into the randomness in malloc allocation, DNS query id allocation, TCP sequence numbers...in virtually anything and everything in OpenBSD.

Last but not the least, OpenSSH is a byproduct of this great OS!

In the next issue:

- Hosting BSD
- Cloud Computing
- Open BSD, NetBSD and FreeBSD as file sharing servers - part2

Next issue of BSD Magazine available in April !



Choosing and installing a Window Manager with FreeBSD

Rob Somerville

One of the many attractive features of BSD is that the end-user is not tied to a particular desktop or windowing environment.

While it is possible to run different shells with other major operating systems, BSD, Linux and Unix are different in that a separate layer (Xorg – the X Windows system) sits between the kernel and the GUI environment. Once Xorg is configured correctly, it is relatively trivial to install a *Window Manager* (WM) – or indeed multiple WM's – if you so choose. At last count at *freshports.org* there were over 60 WM's available for BSD (Table 1), so choice is only limited by your processor and aesthetics.

Set-up and support

Traditionally, setting up X Windows could be very tricky, often due to closed source video drivers or odd monitor configurations. While there are rogue video cards out there that are not natively supported, most cards these days can be persuaded to run in VESA mode. In the author's experience, more modern *Cathode Ray Tube* (*CRT*) and *Liquid Crystal Displays* (LCD) screens will work straight out of the box with Xorg 7.x especially as it now has a fail-safe / auto-config mode. The only difficulty that may arise is with wide-screen configurations or laptops, often these have proprietary hardware or obscure settings that need to be taken into consideration if optimal settings are desired.

As it is possible to overdrive and consequently damage your video hardware, it is always good practice to check that your kit is supported beforehand and confirm optimal resolution, refresh rates, mode line settings etc. especially if you are unsure as to the specification. Modern hardware can probably cope better than older kit, but the wrong setting in

Figure 1. X up and running

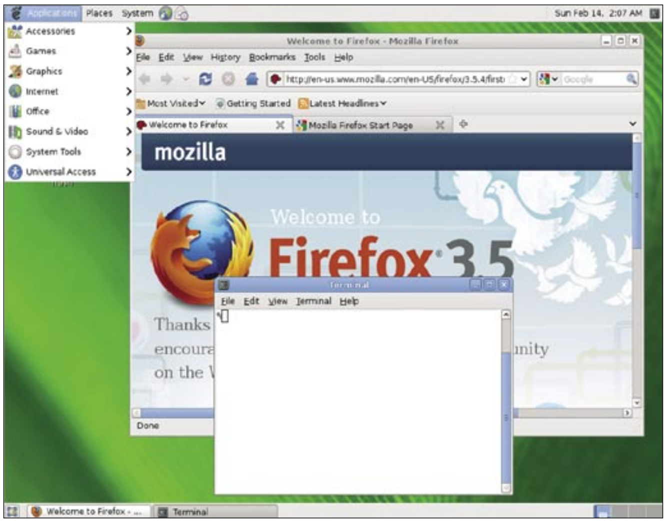


Figure 2. Gnome

Table 1. Window Managers available for BSD

Currently maintained Window Manager ports [Screenshots in bold]	
Aewm 1.2.7_3	ICCCM-compliant window manager based on 9wm
afterstep-stable 2.2.9_1	A stable version of the AfterStep window manager
amaterus 0.34.1_5	A GTK+ window manager
antiwm 0.0.5	A minimalist window manager inspired by Ratpoison
awesome 3.4.3_2	A highly configurable, next generation framework window manager
awesome2 2.3.6_2	A tiling window manager initially based on a dwm code rewriting
blackbox 0.70.1_2	A small and fast window manager for X11R6
ctwm 3.8a_4	An extension to twm, with support for multiple virtual screens, etc
dwm 5.7.2	A dynamic, small, fast and simple window manager
e16 1.0.1.901_1	A very artistic X window manager
echinus 0.3.9	A dynamic window manager for X11 based on dwm
ede 1.2_4	Equinox Desktop Environment
enlightenment 0.16.999.042_4,2	A very artistic X window manager
epiwm 0.5.6_7	Another fast, small, configurable window manager
etoile-azalea 0.2_1	Azalea is a GNUstep window manager
evilwm 1.0.0_1	Minimalist window manager based on 9wm
fluxbox 1.1.1_1	A small and fast window manager based on BlackBox
flwm 1.02_3	The Fast Light Window Manager
fvwm2 2.4.20_2	Popular virtual window manager for X
fvwm95 2.0.43a_4 4	Win95 lookalike version of the fvwm2 window manager
golem 0.0.5_2	Small window manager with themes and plugins
hackedbox 0.8.4_2	Hackedbox is a small and fast window manager based on Blackbox
i3 3.d	An improved dynamic tiling window manager
icewm 1.2.37_3	Window Manager designed for speed, usability and consistency
ion 20020207_2	A window manager with a text-editorish, keyboard friendly interface
jewel 0.12.41_3	Window manager based on aewm++
larswm 7.5.3_2	Tiling Window Manager for X
lwm 1.2.2	A lightweight window manager
matchbox 1.2	Window manager suitable for low-resolution screens

Currently maintained Window Manager ports [Screenshots in bold]	
matwm2 0.1.0	A minimalistic, yet functional window manager for x11
musca 0.9.24	A simple window manager for X
mutter 2.28.0_1	Window and compositing manager based on Clutter
nickleby 1.58	Simple window manager
openbox 3.4.10	A standards compliant, fast, light-weight window manager
oroborus 2.0.18_1	A small and simple GNOME-compatible window manager
pekwm 0.1.11_2,1	Light, Unobtrusive, and configurable windowmanager
phluid 0.0.3_10	A window manager that emphasizes efficiency, speed, and beauty
pwm 2007.07.20_2	A lightweight window manager with emphasis on usability
qvwm 1.1.12_7	Windows 95/98/NT like window manager for X11
ratpoison 1.4.5_1	Simple window manager with no fat library dependencies
sapphire 0.15.8_1	Small window manager
sawfish 1.5.2_2,2	Lisp configurable window manager
scrotwm 0.9.19	A small, dynamic tiling window manager for X11
selectwm 0.4.1_7	A GTK application to let you select a window manager
swm 1.3.4_4	Window manager for low-memory systems
tinywm 1.3_2	Ridiculously tiny window manager
vtwm 5.4.7_3	Twm with a virtual desktop and optional Motif-like features
w9wm 0.4.2_2	A hack of 9wm to give virtual screens
weewm 0.0.2_2,1	Fast and ultra light windowmanager with total keyboard control
windowlab 1.33_2	A small window manager for X11
windowmaker 0.92.0_8	GNUstep-compliant NeXTstep window manager clone
wmfs 201001	A floating and tiling Window Manager From Scratch
wmg 0.15.0_7	Small GTK-based GNOME-compliant window manager
wmii-devel s20090703	A dynamic, minimalist window manager for X11
xfce 3.8.18_10	CDE like desktop with GTK
xfce4-wm 4.6.1_3	XFce 4 window manager
xmonad 0.9.1	Xmonad is a minimalist and tiling window manager for X

the monitor section of xorg.conf could potentially be problematic.

Desktop Environment versus Window Manager

With the plethora of choice available [1], it is possible to install anything from an extreme lightweight such as Ratposion (a mouse-less WM) to a full blown *Desktop Environment* (DE) such as Gnome or KDE. A low specification PC will perform better with a WM rather than a DE.

Security and functionality are prime considerations and a DE will require a lot of additional library support which will not only add to the install time but potentially may add vulnerabilities. While most WM's are *bare bones* and highly customisable via their configuration files, a DE will come pre-configured with lots of additional goodies (such as file managers, printer monitoring utilities etc.) so if anything other than basic functionality is required, a DE may be the best choice.

While it is not best practice to install a GUI in a server environment, occasionally the need may arise (e.g. to run Virtualbox to install a virtual machine O/S from bare metal) a lightweight WM with few *bells and whistles* is useful. Xorg provides TWM as the default WM, and this is sufficient for most purposes.

User interface aesthetics are important as well – some prefer the minimalist approach of Blackbox to the slower but more visually stimulating Afterstep or Enlightenment.

For this article, all software was tested in a virtual machine running FreeBSD 8.0 and Xorg 7.4.

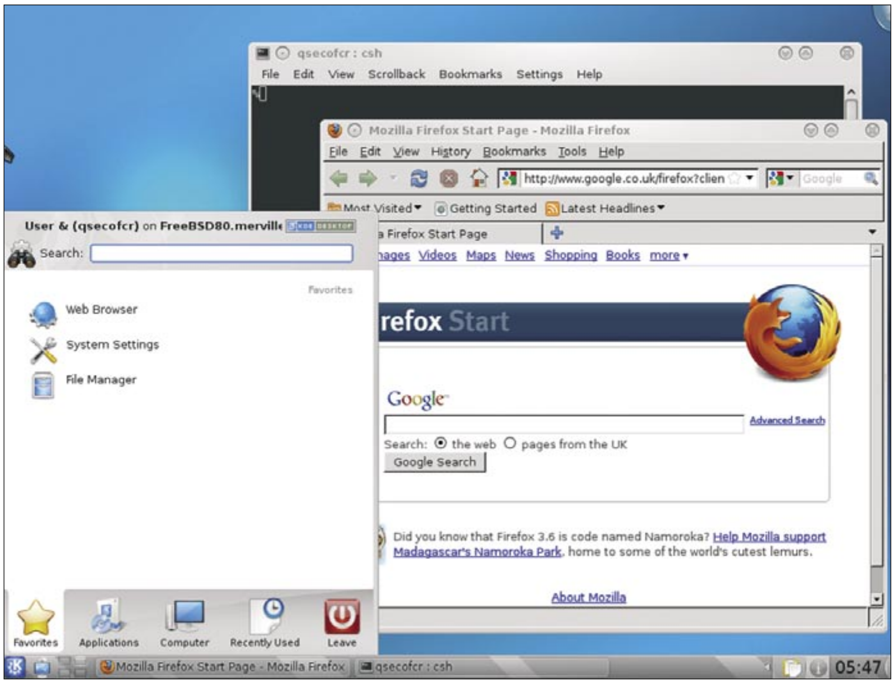

qsecofcr : csh
File Edit View Scrollback Bookmarks Settings Help
Mozilla Firefox Start Page - Mozilla Firefox
User & (qsecofcr) on FreeBSD80.merville
Search:
Favorites
Web Browser
System Settings
File Manager
Favorites
Applications
Computer
Recently Used
Leave
05:47

Figure 3. KDE4

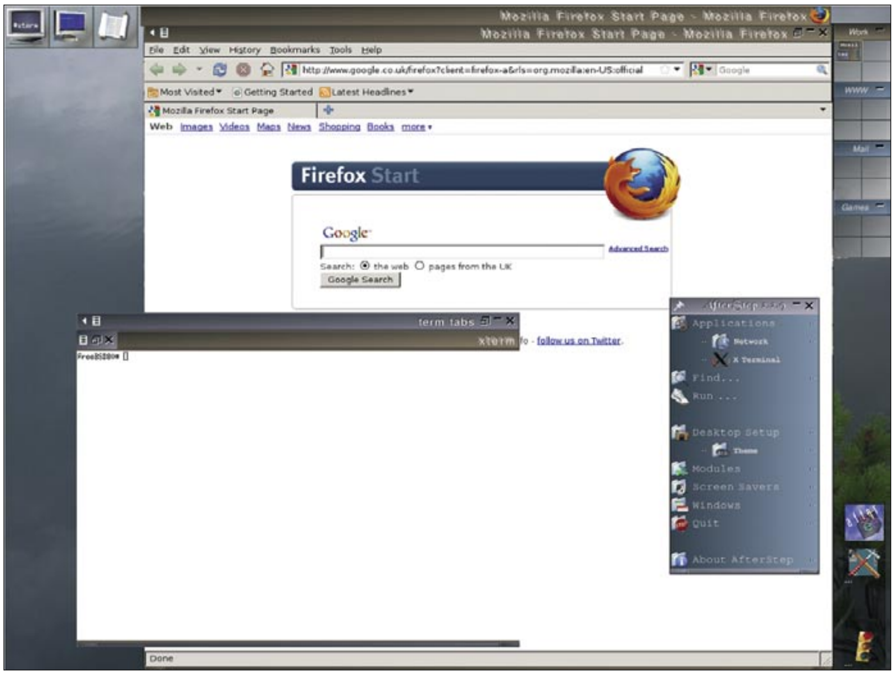


Figure 4. Afterstep

The major players

The heavyweights

These are the full blown desktop environments complete with their own suite of applications, libraries and utilities. It is probably best to install these from the installation DVD via sysinstall due to major package dependencies.

Both these DE's support a wide range of applications, (and apart from having to install some additional libraries) they will often support each other applications as well. For a humorous analysis of the pro's and con's of these DE's see the final smackdown at *linux.mag.com* [2].

- Gnome (see Figure 2) – Traditional desktop with drop-down menus and the Nautilus file manager. More slim-line than KDE, it is the basis for the OpenSolaris desktop. With additional utilities can be themed to look very Mac like. The standard for Redhat Linux.
- KDE (see Figure 3) – Out of the box has a more contemporary styling than Gnome and is the default desktop for SUSE Linux Enterprise. Strong support for educational games and applications.

The middleweights

- AfterStep (see Figure 4) – Based on the Bowman NeXTstep TM clone, this WM has plenty of eye-candy.
- Enlightenment (see Figure 5) – A beautifully crafted WM still heavily under development.

The lightweights

- Window Maker (Figure 6) – Reproduces the elegant look and feel of the NEXTSTEP TM user interface.
- Blackbox (Figure 7) – A minimalist desktop but as a result has a very small footprint.
- TWM – The default WM supplied with Xorg.

Getting Xorg up and running

Xorg needs to be installed either from the FreeBSD DVD or using `pkg_add` (Table 2). If you are running FreeBSD 7.4 or greater, DBUS and HALD are required. If required, ensure the moused daemon is operational and add the following lines to `/etc/rc.conf` as necessary.

```
moused_enable="YES"
dbus_enable="YES"
hald_enable="YES"
```

Manually start the `MOUSED`, `DBUS`, `HALD` services or reboot: As a unprivileged user, start the X server:

```
xinit
```

If all is well, Xorg should start, your mouse should work and you will see a bare-bones X session up and running as shown in Figure 1.

Switch to a console you ran xinit with and press [*ctrl-c*] to terminate (*ctrl-alt-backspace* is disabled in later version of Xorg). If your mouse or display doesn't come up, you will need to generate, test and modify the configuration as appropriate.

As root, run:

```
Xorg -configure
Xorg -config /root/xorg.conf.new -retro
```

You should now see the traditional hatched background and mouse cursor. Copy `xorg.conf.new` file into `/etc/X11/xorg.conf` if you are successful, otherwise refer to the handbook at freebsd.org for more detail on how to proceed.

Installing your DE or WM

Gnome and KDE are supplied as packages on the FreeBSD 8.0 DVD and this was the preferred method of installation to save bandwidth. All other WM's were installed as packages using the following invocation :

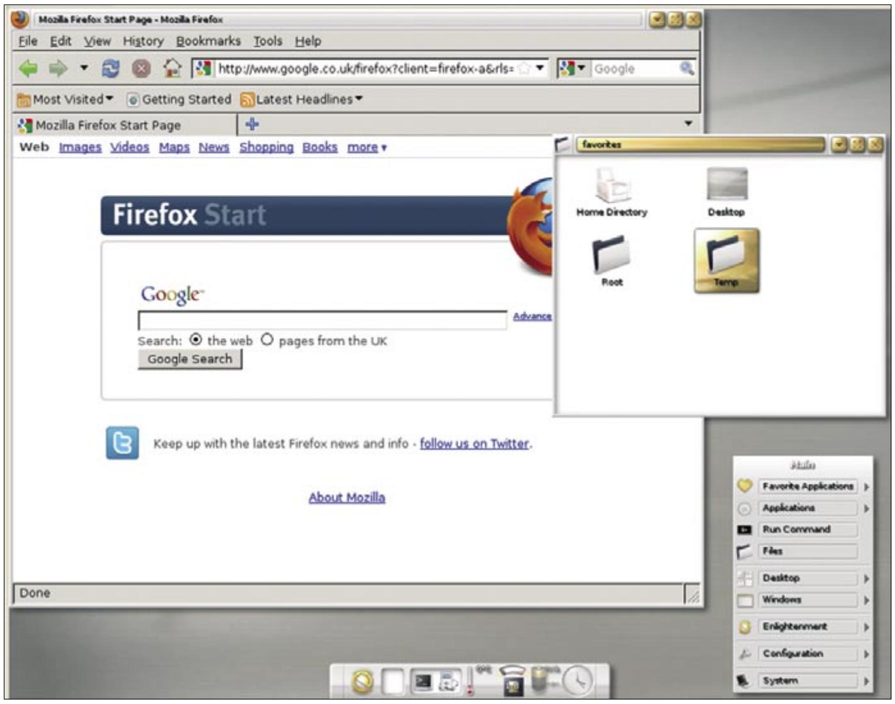

Mozilla Firefox Start Page - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.google.co.uk/firefox?client=firefox-a&rls=
Most Visited Getting Started Latest Headlines
Mozilla Firefox Start Page
Web Images Videos Maps News Shopping Books more
Firefox Start
Google Search
Keep up with the latest Firefox news and info - follow us on Twitter.
About Mozilla
Done
favorites
Home Directory
Desktop
Root
Temp
Favorite Applications
Applications
Run Command
Files
Desktop
Windows
Enlightenment
Configuration
System

Figure 5. Enlightenment

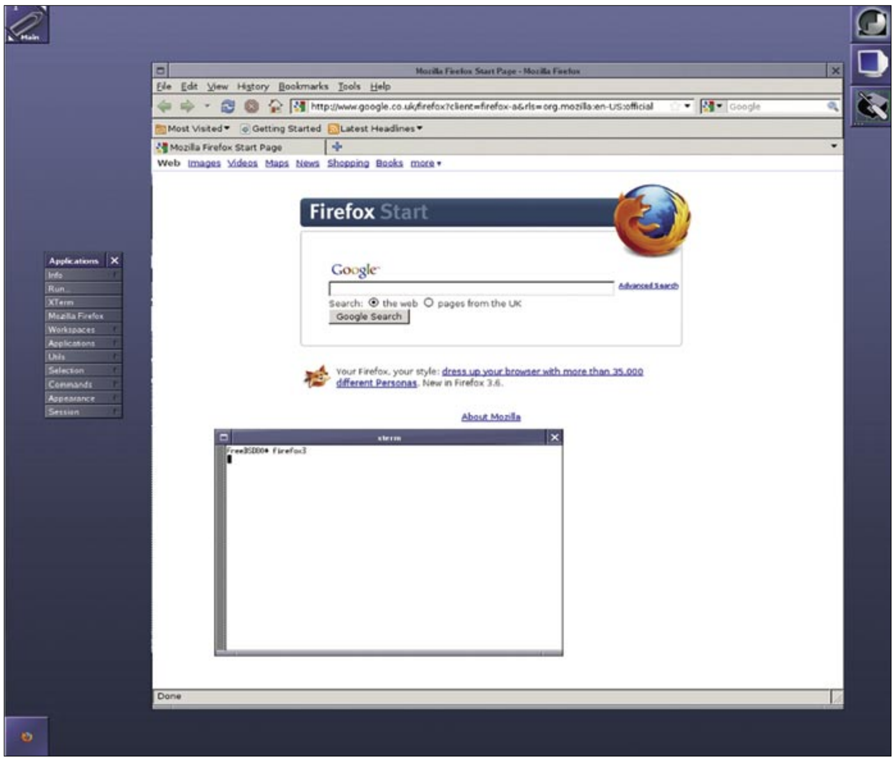

Mozilla Firefox Start Page - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Most Visited Getting Started Latest Headlines
Firefox Start
Google Search
Your Firefox, your style: dress up your browser with more than 35,000 different Personas. New in Firefox 3.6.
About Mozilla
Applications
Info
Run...
XTerm
Mozilla Firefox
Workspaces
Applications
Utils
Selection
Commands
Appearance
Session
xterm
FreeBSD80# firefox3
Done

Figure 6. Window Maker

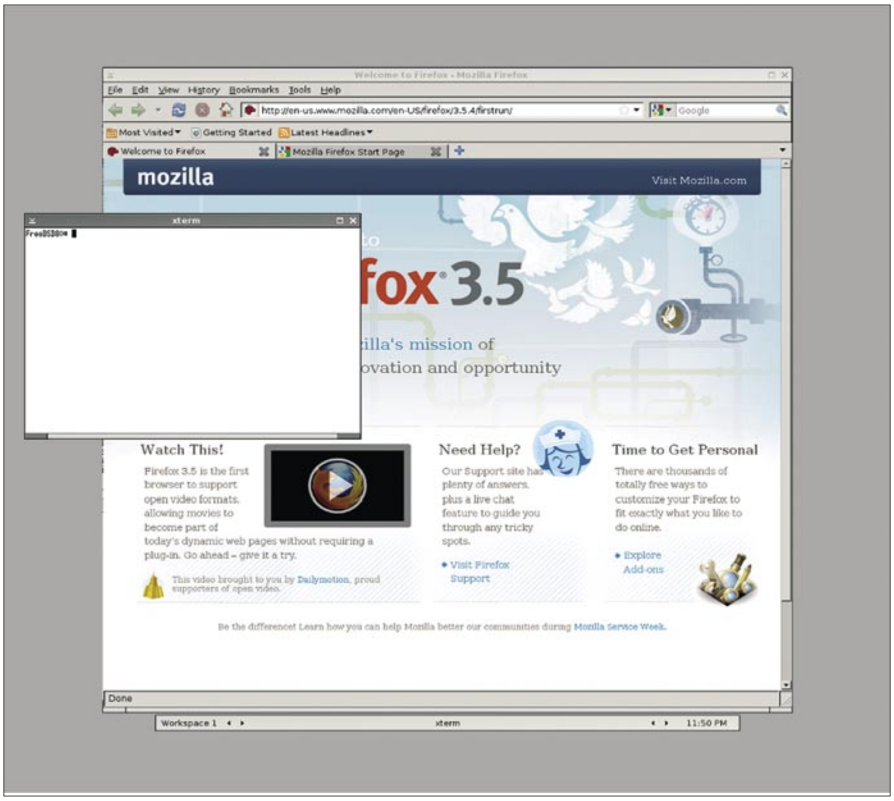
References and notes

- Wikipedia – Comparison of X Window System desktop environments – *http://en.wikipedia.org/wiki/Comparison_of_X_Window_System_desktop_environments* [1]
- Gnome V KDE – The Final Smackdown – *http://www.linux-mag.com/id/7296* [2]
- OpenSolaris – OpenSolaris Desktop *http://hub.opensolaris.org/bin/view/Project+jds/* [3]
- There is no maintainer for this port – N/A

Table 2. xxxxxxxxxx

Installation and xinit commands

Application / Window Manager	pkg_add -r command	xinit / .xinitrc command
Xorg	xorg	N/A
Gnome	Installed via DVD using sysutils	/usr/local/bin/gnome-session /usr/local/bin/gnome-session
KDE4	Installed via DVD using sysutils	/usr/local/kde4/bin/startkde exec /usr/local/kde4/bin/startkde
Window Maker	windowmaker	/usr/local/bin/wmaker exec /usr/local/bin/wmaker
Blackbox	blackbox	/usr/local/bin/blackbox exec /usr/local/bin/blackbox
Afterstep	afterstep	/usr/local/bin/afterstep exec /usr/local/bin/afterstep
Enlightenment	Installed via ports	enlightenment_start exec enlightenment_start



Firefox 3.5 is the first browser to support open video formats, allowing movies to become part of today's dynamic web pages without requiring a plug-in. Go ahead – give it a try.

This video brought to you by Dailymotion, proud supporters of open video.

Our Support site has plenty of answers, plus a live chat feature to guide you through any tricky spots.

There are thousands of totally free ways to customize your Firefox to fit exactly what you like to do online.

Be the difference! Learn how you can help Mozilla better our communities during Mozilla Service Week.

Figure 7. Blackbox

```
pkg_add -r packagename
```

where packagename refers to the WM in question. The only exception was Enlightenment, at time of writing this WM had to be installed via the ports tree as the package appeared broken (Table 2).

```
cd /usr/ports/x11-wm/enlightenment
make
make install
```

Testing the Window Manager

To initially test, the WM can be run via *xinit* e.g. for the Gnome desktop environment:

```
xinit /usr/local/bin/gnome-session
```

See Table 2 for further details.

Creating the .xinitrc file

To facilitate starting the WM using the startx command, the *.xinitrc* file in the users home directory is created:

```
echo "exec /usr/local/bin/blackbox" >
~/.xinitrc
See table 2 for further examples.
```

Login Manager

The login manager presents the user with a GUI immediately after the system boots. Depending on the level of sophistication of the login manager (e.g. Wdm), the user can select what WM to run at login if multiple WM's are installed.

If you require a graphical login manager, gdm is installed as part of Gnome package and can be started at boot by adding the following to rc.conf:

```
gdm_enable="YES"
```

For other login managers, please refer to the relevant man pages.

BSDCan 2010
The Technical BSD Conference
High value. Low cost. Something for everyone.

BSDCan, a BSD conference held in Ottawa, Canada, has quickly established itself as the technical conference for people working on and with 4.4BSD based operating systems and related projects. The organizers have found a fantastic formula that appeals to a wide range of people from extreme novices to advanced developers.

BSDCan 2010 will be held on 13-14 May 2010 at University of Ottawa, and will be preceded by two days of Tutorials on 11-12 May 2010.

There will be related events (of a social nature, for the most part) on the day before and after the conference.

http://bsdcan.org/

BSDCan 2010

BSD Live Desktops

Jesse Smith

The BSD family has long held a well deserved reputation for being superb server operating systems. However, OS X aside, it's it's not very often we hear about BSD on the desktop. That's too bad, because many of the things which make BSD a perfect solution in the server room are also great characteristics to have in a desktop system. When I hear “BSD” I think of stability, speed, grace under heavy work loads and a practical immunity to most viruses. Who wouldn't want those traits in their desktop? It's that sort of thinking which has lead to projects such as PC-BSD (http://pcbsd.org), which took FreeBSD and placed a desktop layer over it. And FreeBSD isn't the only member of the family being dressed up and displayed to the masses. Recently NetBSD and OpenBSD have also been getting friendly new looks via the Jibbed (http://www.jibbed.org) and GNOBSD (http://gnobsd.sri-dev.de) projects respectively. Both projects take the basic system and add a user-friendly desktop on a live disc. This makes exploring OpenBSD and NetBSD an easier task for people who might not otherwise test drive these powerful operating systems. Last week Zafer Aydogan, founder of Jibbed, and Stefan Rinkes, founder of GNOBSD, agreed to talk about their projects, themselves and BSD.

BSD Mag: To start, could you please tell us a little about yourself. Where you are from and how you got started with BSD?

SR: I'm 23 years old and live in Kirchheim, near Munich in Germany. I had my first experiences with OpenBSD during my apprenticeship as an IT specialist about two years ago.

ZA: I'm living in Kiel, Germany, where I was also born, but as you can see from my name I'm Turkish. I was raised bilingual.

I got in touch with FreeBSD in 1999 and moved to NetBSD in 2000. I've never used OpenBSD. In my day job, I work as a software engineer in a small company.

BSD Mag: Why did you decide to start your project?

ZA: NetBSD does not create a live CD during the release process, therefore there are no official live CDs. The ones available are outdated. I thought there would be a demand for NetBSD live CDs that are up to date, giving, especially, non-NetBSD users the opportunity to get in touch with the OS without installing it.

One of the characteristics of NetBSD is their friendly and knowledgeable community. Building and customizing the live CD was not too difficult, since there was already a script for building live CDs in pkgsrc. I just built a framework around it, to keep it simple and to be able to release regularly.

SR: I'm one of those *learning-by-doing* guys, so GNOBSD was the perfect way to learn more about OpenBSD, the installer, programming (shell and Ruby) and how to design GUIs. GNOBSD was a learning-project and I learned a lot.

I also thought it would help some people to try OpenBSD and to test to see if all of their hardware is detected and working properly.

BSD Mag: What sort of feedback have your received on the project? Have people offered suggestions, bug reports, feature requests, assistance?

SR: The feedback I received was more positive than negative. Some people even offered to set up a mirror for GNOBSD. I'm looking forward to getting more e-mails with feedback.

ZA: I have received mostly positive feedback. People are happy if they can run NetBSD on their hardware. Negative feedback comes from users that either expect something different or are unable to run the CD. I'm trying to implement suggestions, where possible and useful.

BSD Mag: OpenBSD and NetBSD are known for their security and flexibility on servers, do you think they also make for good desktop systems?

ZA: Definitely. NetBSD has made remarkable progress in being a desktop system. It still needs some effort and patience during setup, but it is possible to have a decent desktop system including Flash, being able to run Java applications and Cisco VPNs to your company network.

SR: Of course. I think it is important to use a secure OS as a desktop system. You can use OpenBSD for a secured office workstation, it has all you need. With a stability which is hard to find elsewhere.

BSD Mag: GNOBSD and Jibbed are great ways to experiment with OpenBSD and NetBSD. Will there be future releases of these projects? Are there any new features planned?

SR: There is the idea to provide code, scripts and documentation, so that everyone can build his/her own version of GNOBSD. As easy as GNOBSD, but customized to individual needs.

ZA: I'm currently preparing Jibbed-5.0.2, which will be released as soon as NetBSD-5.0.2 has been announced. The most significant change on the CD will be the switch to modular Xorg, which is more up to date and can provide support for more graphics cards than Xorg in the base system. I've also added a couple of new applications. The release after that will be 5.1. I'm experimenting with Gnome as a new window manager and I'm also preparing a memory-stick-version of Jibbed. The benefits of having writeable media makes it a portable NetBSD system on stick. Additionally, I have recently started to code a graphical installer using GTK2. It was bugging me for a long time, not having one. Depending on my spare time, you can expect a working version in the next 12 months.

BSD Mag: Are you currently working on any other projects?

SR: Just some small stuff, like microcontroller programming and porting some stuff to OpenBSD. And I'm currently studying for a successful conclusion of my apprenticeship. I'm sure there will be some new projects when the exams are over.

ZA: No, I'm not. But, I would like to thank all mirrors, who are generously providing huge

amounts of bandwidth every month and of course all NetBSD developers for making one of the best operating systems in the world.

BSD Magazine would like to thank Herr Rinkes and Herr Aydogan for taking the time to talk about their projects. Your author has had a chance to play with both systems and they do indeed provide a easy way to explore a BSD system, lowering the bar for new adoption. The BSD community will no doubt benefit from their work.

About the author

Jesse Smith is a system administrator and programmer by training, an open source advocate by choice and a writer at heart. When he's not working with computers, he loves spending time with his family and enjoying the natural beauty of his native Canada.

BSD Goes to the Office: Can BSD compete in a real life consulting workplace?

Mike Bybee
Consultant, Fujitsu America

There are many articles that expound on the success of Linux as desktop, and quite a few accounts of using a Linux desktop in this case or that case.

There are fewer articles regarding a BSD based system as a desktop, largely due to its status as solid back office system. BSD quietly powers a great deal of devices in its many variations and incarnations, from firewalls and mail servers to routers and switches. Running BSD as a desktop is still relatively rare, and it is a far less publicized desktop than its distant cousin, Linux. What fame it has relates mostly to the common base it shares with Darwin, the base for Apple's popular OSX system.

I undertook an experiment with the support of my employers to determine the viability of a BSD desktop in a real world high pressure consulting engagement. The life of a consultant requires absolute attention to the requirements of a client, so there can be no compromises made on their side to allow for any incompatibilities or shortcomings. This differs from the usual article of this type by providing a systems administration and consulting perspective instead of a journalist or home user perspective.

For this experiment I chose to use Sun Virtual Box to run a PC-BSD 8 Release Candidate guest operating system. Sun Virtual Box was chosen largely due to the ease of use and support for a wide variety of guest and host operating systems. PC-BSD 8 was chosen as it provides an excellent end-user experience with a minimum of steps, and is based on the new FreeBSD 8. The hardware I'm running this on is an older Fujitsu LifeBook E8110 with 1GB of RAM. The primary OS is the venerable Windows XP, still the darling of the corporate world. The performance is poor but tolerable, and there is typically noticeable swapping during regular usage.

Here you can see my standard corporate desktop, and inset the new guest OS that will soon supplant it. From the task manager, you can see that it produces a very minimal load on the host system.

The first step was to install Sun Virtual Box 3.12 on Windows XP. The installation was quick and straightforward. I then created a guest system, choosing FreeBSD as the guest OS. I set the memory to a mere 400MB, enabled acceleration, and created a 20 GB virtual disk based on dynamic allocation. I then inserted a PC-BSD 8 install DVD and started the guest OS.

PC-BSD has an attractive graphical installer, which provided a good guide for the installation process. I chose to manually set up the partitions, and enabled whole disk encryption for `/usr` (based on GELI). There was no prompt to allow me to set up a manual password for this disk, but this was easy enough to configure later. I chose to install all of the default packages, including Firefox, Open Office, VLC, Pidgin and more, and then let it run.

The install ran for about 30 minutes without presenting too much of an issue for my standard tasks. At this point, my machine was swapping a bit as 1GB of RAM is not really enough even for standard Windows XP to run comfortably. There was some lag on the desktop, but it was really only noticeable in Outlook.

After a reboot of the guest machine, I was confronted with a minor bug in the install process; the system was attempting to mount `/usr` with an invalid label. This has been fixed in the installer code and shouldn't cause any further issues. I rebooted once the error was corrected and it proceeded into a graphical screen where I could choose my X driver and resolution. As of this writing, the native Virtual Box drivers are not included, so I had to set it initially to VESA. The KDE 4.3.4 desktop popped up, fully populated with the apps I had chosen, and response was surprisingly good. I installed the Virtual Box drivers from `/usr/ports/emulators/virtualbox-ose-additions`, set `vboxguest_enable="YES"` in `/etc/rc.conf`. Restarting without

X running, and then executing Xorg -configure handled the rest of the display setup.

Once the Virtual Box drivers were turned on it was very smooth, attractive, and seamless. I was even able to enable the KDE compositing effects via XRender while still getting decent responsiveness from the GUI. At this point, I was ready to begin configuring the new desktop.

The Xmarks utility provided quick and painless synchronization of my bookmarks and (optionally) stored passwords from my primary desktop to my PC-BSD desktop. I immediately switched off my desktop browser in favor of the guest and was impressed that nearly every site I routinely access worked fine.

The client I am at makes great use of Microsoft SharePoint for collaboration and document storage, and it doesn't integrate with Firefox as tightly as with Internet Explorer. All the same, I was just prompted for my Active Directory password and a handler app for the various files. Open Office responded quickly and smoothly, and the experience of opening, reading, and editing documents on Share Point was acceptable. Open Office even handled password-protected documents properly.

The next major hurdle was handling remote Microsoft Windows servers. Many clients use Windows servers and they typically need to be accessed via Remote Desktop. The PC-BSD Software Manager (conveniently accessible via a desktop or menu shortcut) provided me quick access to the PBI repository at *http://pbidir.com* as well as integrating the installation and version management features. I was able to find and download the PBI package for Remote Desktop quickly and install it without any fuss whatsoever. Testing showed that it worked quite well accessing remote Windows hosts, and this was another feature that I could transition off my primary desktop.

The native tools performed well for the Unix management tasks. A great deal of my time is spent administering Unix hosts with SCP, SSH, and various X11-based utilities. Under Windows this is accomplished with a variety of tools – PuTTY, WinSCP, and of course an X11 client are all commonly used. PuTTY is available as a PBI in the event you need it, but in my experience I found it was faster and easier to use the native tools under all circumstances. Likewise scripting and editing files is easy via Vim or Kate, with Emacs available as a PBI package.

Instant Messenger functionality is crucial in a modern distributed workplace, and some corporations have a specific list of approved instant messenger clients. Normally I would use Pidgin and a plug-in such as SIPE if needed. Since WINE does

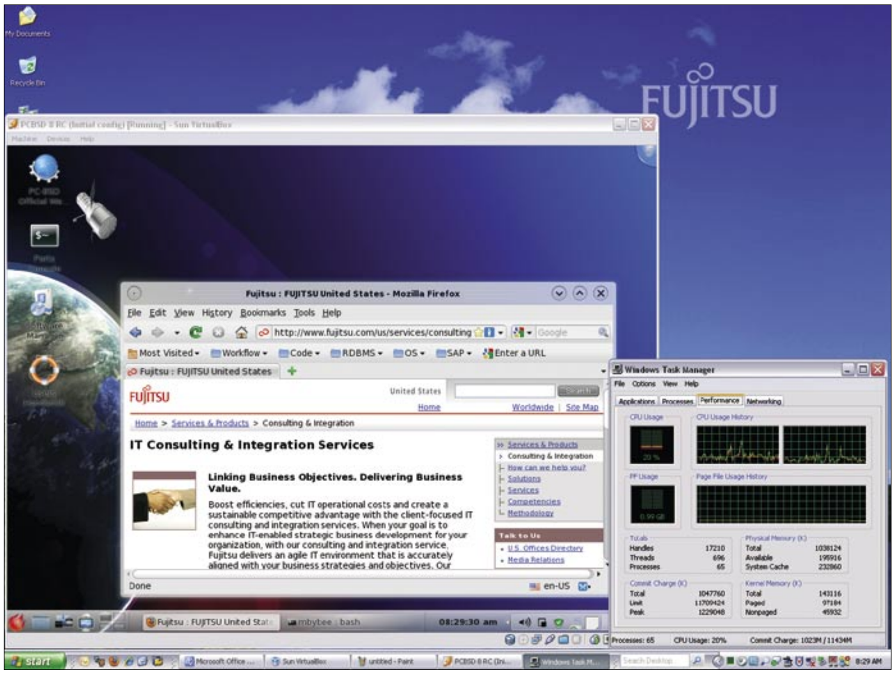
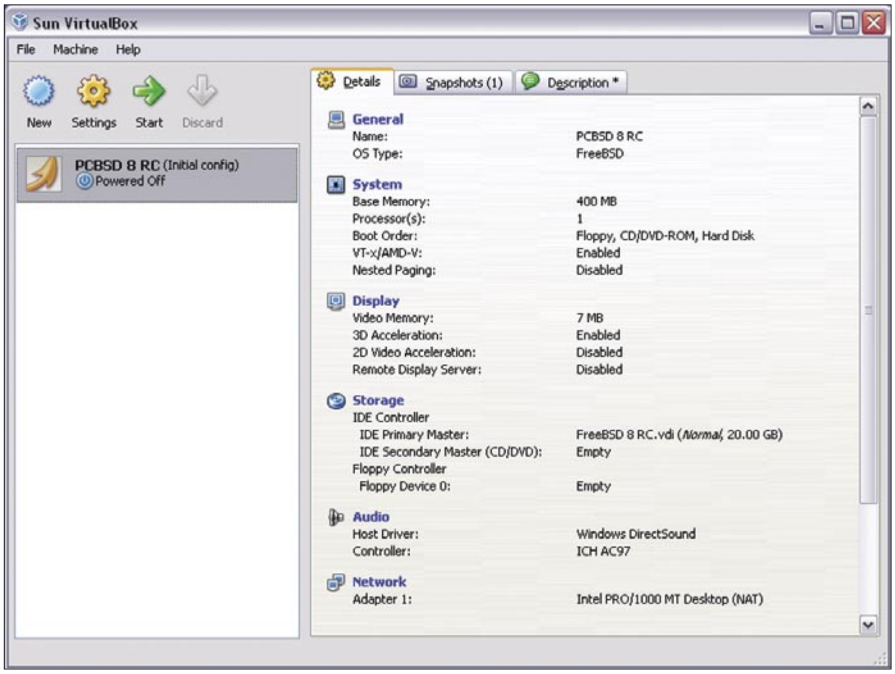

FUJITSU
Fujitsu : FUJITSU United States - Mozilla Firefox
IT Consulting & Integration Services
Linking Business Objectives. Delivering Business Value.
Windows Task Manager

Figure 1. Windows XP running PC-BSD 8 as a guest OS

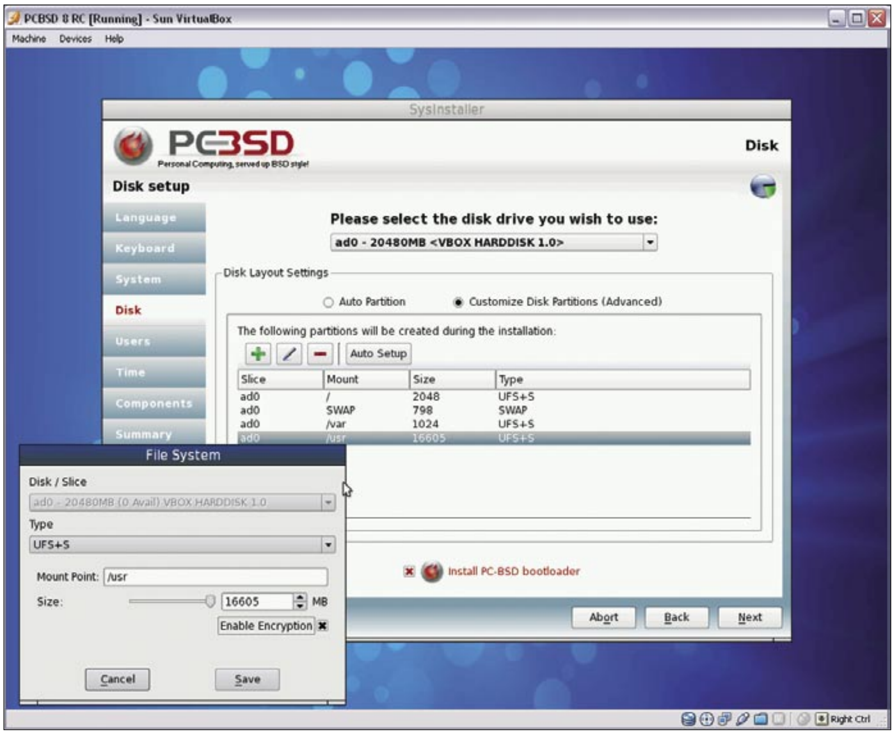

Sun VirtualBox
File Machine Help
New Settings Start Discard
Details Snapshots (1) Description *
PCBSD 8 RC (Initial config)
Powered Off

General	
Name:	PCBSD 8 RC
OS Type:	FreeBSD
System	
Base Memory:	400 MB
Processor(s):	1
Boot Order:	Floppy, CD/DVD-ROM, Hard Disk
VT-x/AMD-V:	Enabled
Nested Paging:	Disabled
Display	
Video Memory:	7 MB
3D Acceleration:	Enabled
2D Video Acceleration:	Disabled
Remote Display Server:	Disabled
Storage	
IDE Controller	
IDE Primary Master:	FreeBSD 8 RC.vdi (*Normal*, 20.00 GB)
IDE Secondary Master (CD/DVD):	Empty
Floppy Controller	
Floppy Device 0:	Empty
Audio	
Host Driver:	Windows DirectSound
Controller:	ICH AC97
Network	
Adapter 1:	Intel PRO/1000 MT Desktop (NAT)

Figure 2. Sun Virtual Box settings

not yet provide good support for this app and we were prohibited from using any other client, I fell back to using the web-enabled version.

Many tasks involve managing large enterprise databases such as Oracle. The OEM Grid Control tools functioned without issue, and it is even possible to install Oracle Express Edition for Linux via the built in Linux Compatibility layer under BSD, though I didn't have the time to complete that for this test. Management


PCBSD 8 RC [Running] - Sun VirtualBox

Disk setup

Please select the disk drive you wish to use: ad0 - 20480MB <VBOX HARDDISK 1.0>

Disk Layout Settings: ( ) Auto Partition [✓ Customize Disk Partitions (Advanced)]

The following partitions will be created during the installation:

Slice	Mount	Size	Type
ad0	/	2048	UFS+S
ad0	SWAP	798	SWAP
ad0	/var	1024	UFS+S
ad0	/usr	16605	UFS+S

File System

Disk / Slice: ad0 - 20480MB (0 Avail) VBOX HARDDISK 1.0

Type: UFS+S

Mount Point: /usr

Size: 16605 MB

Enable Encryption

Figure 3. Advanced Disk Layout

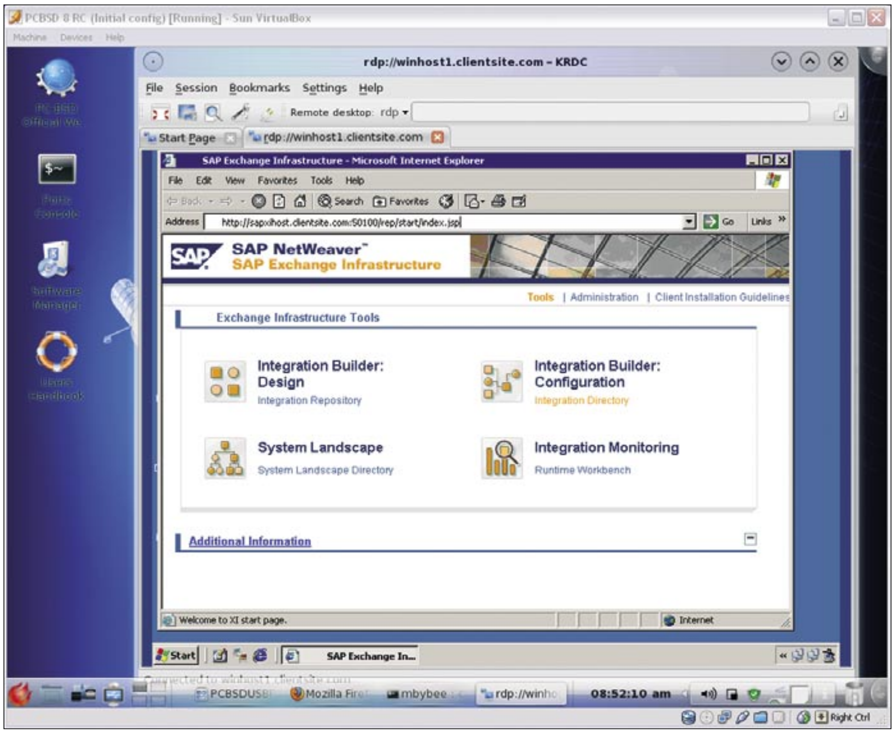

PCBSD 8 RC (Initial config) [Running] - Sun VirtualBox
rdp://winhost1.clientsite.com - KRDC
File Session Bookmarks Settings Help
Start Page
rdp://winhost1.clientsite.com
SAP Exchange Infrastructure - Microsoft Internet Explorer
http://sapxihost.clientsite.com:50100/rep/start/index.jsp
SAP NetWeaver
SAP Exchange Infrastructure
Tools | Administration | Client Installation Guidelines
Exchange Infrastructure Tools
Integration Builder: Design
Integration Repository
Integration Builder: Configuration
Integration Directory
System Landscape
System Landscape Directory
Integration Monitoring
Runtime Workbench
Additional Information
Welcome to XI start page.
Internet
SAP Exchange In...
08:52:10 am

Figure 4. Remote Desktop under PC-BSD 8

On the 'Net

- *http://www.virtualbox.org/*
- *http://wiki.freebsd.org/VirtualBox*
- *http://sipe.sourceforge.net/*
- *http://www.pcbsd.org/*
- *http://www.chiark.greenend.org.uk/~sgtatham/putty/*
- *http://pbidir.com/*
- *http://www.xmarks.com/*
- *http://en.wikipedia.org/wiki/Geli_%28software%29*
- *http://www.freebsd.org/doc/handbook/disks-encrypting.html*

of DB2 was also easy. I did run into some road bumps managing Microsoft SQL servers. The easiest method for managing SQL servers was simply to log in via Remote Desktop.

My take away from this experience is that BSD is now capable of being used in a professional consulting environment, and provides the same levels of supportability and security that we expect from our servers in an extremely inexpensive package. PC-BSD 8 performed better than I had hoped, especially when running on so little memory. It is telling that a guest OS running in only 400 MB of RAM was able to outperform a bare metal host OS with full access to the entire 1 GB. Sun Virtual Box also ran very well, and did a great job running a guest OS without impacting the host excessively. Running in LiveUSB mode with full access to the host, PC-BSD gave my laptop a new lease on life. As a replacement OS it would probably extend the service life of this laptop by an additional year or two, and with a lower operating cost.

SAINT®

Integrated Vulnerability Assessment and Penetration Testing

Examine, expose, and exploit your vulnerabilities before an attacker does

Examine your network with the SAINT® vulnerability scanner, and **expose** the areas where an attacker could breach your network. Then, take the next step and **exploit** the vulnerability. This allows you to focus on the high-severity vulnerabilities and provides a starting point for prioritizing remediation efforts.

What SAINT can do for you –

- ✔ Lets you exploit vulnerabilities found by the vulnerability scanner
- ✔ Show you if your network is compliant with PCI security standards. SAINT is a PCI DSS approved scanning vendor.
- ✔ Show you where to begin remediation—with the exploitable vulnerabilities
- ✔ Let you scan and exploit both IPv4 and IPv6 addresses
- ✔ Give you cross references to CVE, CVSS, IAVA, OSVDB, and SANS/FBI Top 20
- ✔ Allows you to generate colorful and easy-to-read reports

Get free white papers and more information at **www.saintcorporation.com**

Contact SAINT's sales team at 1-800-596-2006 x0119 or sales@saintcorporation.com

Extreme Network Security

Orion II iX-N4236

Powerful *4U* Orion II Storage Series

- ✓ Outstanding performance
- ✓ Excellent cooling efficiency
- ✓ Up to 72 TB in 4U, unparalleled storage density

To order today call: **1-800-820-BSDi**

Notable features include:

- Dual Intel® 64-Bit Socket 1366 Quad-Core or Dual-Core, Intel® Xeon® Processor 5500 Series
- 4U Storage Server Chassis with up to 72 TB storage capacity
- 36 x 3.5 Hot-Swap SAS/SATA HDDs (24 front side + 12 rear side)
- 1400 W (1+1) Redundant High Efficiency Power Supply (Gold level 93%+ power efficiency)
- Dual Intel® 5520 chipsets with QuickPath Interconnect (QPI) up to 6.4 GT/s
- Up to 144GB DDR3 1333/1066/800 MHz ECC Registered DIMM/24 GB Unbuffered DIMM
- 2 (x16) PCI-E 2.0, 4 (x8) PCI-E 2.0 (1 in x16 slot), 1 (x4) PCI-E (in x8 slot)
- Intel® 82576 Dual-port Gigabit Ethernet Controller

iXsystems Introduces the Orion II 4U Storage Solution

The iX-N4236 boasts energy efficient technology and maximum, high density storage capacity, creating a 4U powerhouse with superior cooling.

The Orion II has **thirty-six hot-swappable SAS/SATA drive bays**, providing 50% more storage density than its predecessor. By delivering high-end storage density within a single machine, iXsystems cuts operating costs and reduces energy requirements.

Storage sizes for the iX-N4236 are customizable, with 250GB, 500GB, 750GB, 1TB, and 2TB hard drives available. For environments requiring maximum storage capacity and efficiency, 2TB Enterprise-class drives are available from Western Digital®, Seagate®, and Hitachi. These drives feature technologies to prevent vibration damage and increase power savings, making them an excellent choice for storage-heavy deployment schemes.

Powerful Intel® Xeon® 5500 Series Processors have a light footprint, while creating a perfect environment for intense virtualization, video streaming, and management of storage-hungry applications. Energy efficient DDR3 RAM complements the other power saving components while still providing 18 slots and up to144GB of memory overall.

100% cooling redundancy, efficient airflow, and intelligent chassis design ensure that even under the heaviest of workloads, the Orion II remains at an optimal temperature, while still drawing less power than other servers in its class. With a 1400 W Gold Level (93%+ efficient) power supply, the entire system works together to efficiently manage power draw and heat loss.

For more information or to request a quote, visit:
http://www.iXsystems.com/Orion2

iX systems™

intel® Xeon® inside™

Powerful. Intelligent.