MAGAZINE BSD

FOR NOVICE AND ADVANCED USERS

BSD SECURITY

INSIDE

LARGER SCALE FREEBSD

DRAGONFLYBSD: TESTING HAMMER DEDUPLICATION ON REAL-WORLD DATA

PC-BSD'S NEW CONTROL PANEL

USING POSTGIS TABULAR AND GEOGRAPHIC DATA WITH FREEBSD

COLLECTD - SYSTEMS STATISTICS COLLECTION DAEMON

USING MEMCACHED FOR HIGH SCALABILITY WEB SERVICES

BUILDING A COMPLETE INTRUSION DETECTION SYSTEM WITH SNORBY ON BSD

LDAP AUTHENTICATION AND AUTHORISATION OF UNIX USERS UNDER OPENBSD

FULL DISK ENCRYPTION ON FREEBSD

WHAT IT TAKES: OPEN SOURCE CERTIFICATION, PART II

EUROBSDCON 2011: INTERVIEW WITH PAUL SCHENKEVELD

VOL.4 NO.7
ISSUE 07/2011(24)
1898-9144

Titan FreeNAS™ Storage Appliance: You are the Cloud

With a rock-solid FreeBSD® base, Zettabyte File System support, and a powerful Web GUI, FreeNAS™ pairs easy-to-manage software with world-class hardware for an unbeatable storage solution.

JBOD Expansion Available

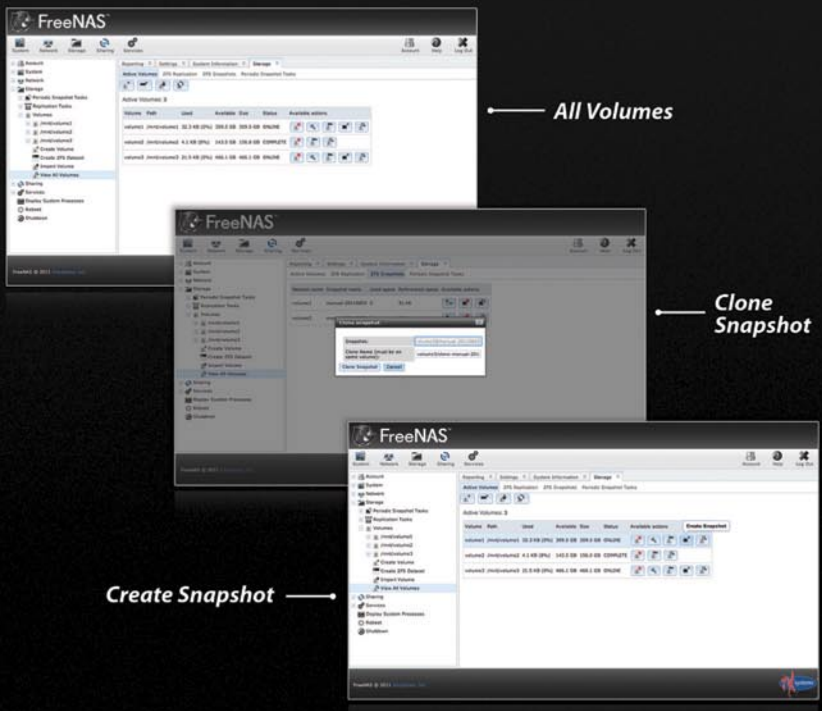
Titan FreeNAS™ 2U Pro System

Titan FreeNAS™ Mini

Storage. Speed. Stability.

For a small business or home office, the Titan FreeNAS™ Mini provides compact storage with small energy demands. For larger operations that use significantly more data, the Titan FreeNAS™ Pro, equipped with the Intel® Xeon® Processor 5600 Series, supports Fusion-io's Flash Memory cards and is available in rackmount form factors from 1U to 4U. Paired with optional JBOD expansion units, the Titan FreeNAS™ Pro offers excellent capacity at an affordable price.

For more information on the **FreeNAS™ Mini** and **2U Pro System**, or to request a quote, visit: **http://www.iXsystems.com/FreeNAS**.



TITAN FREENAS™ MINI KEY FEATURES

- One Dual-Core Intel® Core™ i3 Processor
- 4 Hot-Swap Drive bays - Up to 12TB of Data Storage Capacity
- Software RAID
- 8GB of DDR3 Memory in a 2 x 4GB Layout
- 1 x 1GbE Network Interface (Onboard)
- Single 120W High-Efficiency Power Supply

TITAN FREENAS™ 2U PRO SYSTEM KEY FEATURES

- Supports One or Two Quad-Core or Six-Core, Intel® Xeon® Processor 5600 Series
- 12 Hot-Swap Drive Bays - Up to 36TB of Data Storage Capacity*
- HBA or RAID Controller with Battery Back-up Unit
- Minimum 12GB of DDR3 ECC Registered Memory in a 3 x 4GB Layout Upgradeable to 192 GB
- Up to 4.48TB of Fusion-io Flash Memory
- 2 x 1GbE Network interface (Onboard) + Up to 4 Additional 1GbE Ports or Single/Dual Port Chelsio or Intel® 10GbE Network Cards

JBOD expansion is available on the 2U Pro System

** 2.5" drive options available; please consult with your Account Manager*

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

Powerful.
Intelligent.

Dear Readers,

It's a common understanding in the BSD community that BSD systems are one of most secure OS's you can have. It seems so obvious that sometimes we don't see the need of even saying it – which is why it's quite a long time since we published a security article. This month we have decided to change that a bit – and you can see the result in the articles written by Michael Shirk and Matthieu Bouthors, who will help you secure your computer even better.

Of course that's not everything we have to offer in this issue. As always, you will find a lot of interesting content in Developers Corner, How Tos and Let's Talk sections, as well as an interview with Paul Schenkeveld – one of the organisers of this year's EuroBSDcon.

We are waiting for your comments, replies, ideas and suggestions. Every bit of feedback helps us improve the magazine :).

Thank you!

Zbigniew Puchciński
Editor in Chief
zbigniew.puchcinski@software.com.pl

MAGAZINE BSD

Editor in Chief:
Zbigniew Puchciński
zbigniew.puchcinski@software.com.pl

Contributing:
Josh Paetzel, Justin C. Sherrill, Dru Lavigne, Rob Somerville, Bill Harris, Diego Montalvo, Michael Shirk, Antoine Jacoutot, Mathieu Bouthors, Jim Brown

Proofreaders:
Sander Reiche, Christopher J. Umina, Jeff Loupe

Special Thanks:
Denise Ebery

Art Director:
Ireneusz Pogroszewski

DTP:
Ireneusz Pogroszewski

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

CEO:
Ewa Dudzic
ewa.dudzic@software.com.pl

Production Director:
Andrzej Kuca
andrzej.kuca@software.com.pl

Executive Ad Consultant:
Ewa Dudzic
ewa.dudzic@software.com.pl

Advertising Sales:
Zbigniew Puchciński
zbigniew.puchcinski@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*

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

The editors use automatic DTP system AUPUS

Mathematical formulas created by Design Science MathType™.

Contents

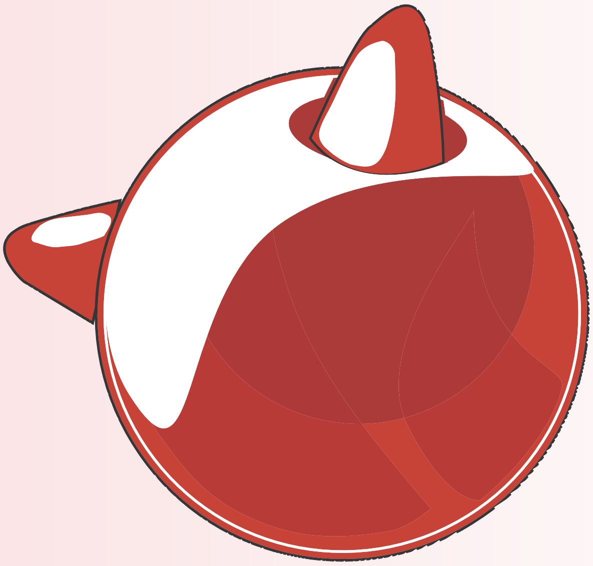
Developers Corner

How To's

Security

Let's Talk

Interview



Larger scale FreeBSD

I've often joked that it's quite easy to manage one FreeBSD machine or one thousand, but if you have ten machines it can be quite a bit of work.

There's more than a grain of truth to this. Stock FreeBSD is a very source-centric system. Third party software can be compiled from source using one of the best frameworks out there, updates to the OS can be done by recompiling the system from source, FreeBSD is self-hosting, and while there are binary tools for many things now, that wasn't always the case and the inertia of FreeBSD being a source based system where compiling is expected carries on to this day. That's not the only issue you'll see running into large-scale deployments. The installer is designed so you can put a CD into a system and do an interactive install. The base services in the OS have to be configured by hand to authenticate against various directory services such as LDAP or NIS.

My goal in this article is to raise awareness of some of the tools that can make your life easier as you scale out your FreeBSD deployments. Consider this a high-altitude bird's eye view, in which I'll be focusing more on the tools that are out there as well as the 'why' to use them. Followup articles will use this as a roadmap of sorts and will provide more of a tactical nitty-gritty howto of the various components that can help with large deployments, focusing more on the *how*.

Conceptually speaking, you can divide the problems that large numbers of FreeBSD machines will give you into several catagories:

- Installation
- Updates
- 3rd party software packaging and distribution
- Configuration management
- User Authentication

Installation

So you've decided to add a FreeBSD machine to your stable. You toss the machine on a bench, grab the latest copy of FreeBSD from ftp.freebsd.org and boot up a CDROM to do an interactive install. This works very well for you, and your FreeBSD machine is such a terrific addition to your network that soon you find use for another system. This continues ad-naseum until you one day realize that this practice is very tedious (You can now navigate the installer with your eyes closed), it's resource intensive, and unless you're the sort of person that keeps detailed notes around, it's easy to have slight differences between installs (Why is `/var` on www3 4GB and it's 6GB on www4?) There's another thing you'll notice too, the installer hasn't kept up with the OS in terms of functionality. Many of the ways you can install a system, such as ZFS on root using GPT labelling, or installing to a geom_mirror, simply aren't options in the installer you'll get with stock FreeBSD. There are many many different options available here to make this process easier, less time consuming, more repeatable, and to unlock some of the advanced functionality in FreeBSD. Many modern systems are capable of being booted from the network via PXE. Once the system is booted this way a variety of options are available to install an OS on the system. The installer can be scripted, you can use pc-sysinstall, the system can run a shell script or other program that has been customdeveloped to do the install, or a raw image can be copied directly to the disks with a utility like dd. Another option is to roll an in house CD or DVD that does a customized install. Whatever the route you take, you'll be really happy you did something as you image server No.58 and get it ready to be inserted into your web-cluster.

Updates

A time honored tradition in running FreeBSD is to use csup or cvsup to update the system sources, then a combination of make commands and mergemaster to compile then update the OS. This method works quite well for a very small number of systems, but you'll soon run into scalability issues as the number of machines you have grows. A few techniques are available to ease this. A build machine can compile an image that can then be shared out to multiple machines via NFS. sysutils/etcmerge can be used instead of the stock mergemaster to ease updating configuration files. There is also a binary updating tool that is available called freebsd-update that in many cases can apply updates to a machine with no compiling at all.

3rd party software

There is a lot of software available for FreeBSD, and the reality is, that for many applications you'll be adding software to your systems to perform specific tasks. As a very source-based system, using the ports tree is attracyive for installing software. This very quickly turns into a quagmire, as you attempt to keep ports tree synchronized so all of your machines have the same version of apache on them, and one day you realize that grabbing a ports tree and compiling apache from source just isn't a viable way to get server No.46 running in your cluster.

Your headaches only increase as the PCI compliance folks insist on you running the latest version of PHP and Apache, and you find a need to update software on a regular basis. There are many ways to do software and package management, and depending on the problems that your deployment presents you'll end up with a solution that fits your needs. The basis for many solutions is going to be a tool called tinderbox, vailable in the ports tree under ports-mgmt/tinderbox. The binary packages emitted by this tool can be combined with updating tools such as portmaster, or with jail image creation scripts, to create completely binary images that can be rolled out cookie-cutter style to boxes that provide services.

Configuration management

At first applying configuration changes to one box isn't much hassle at all. You log in and change a thing or two with vi. But it's not very long before updating the nameserver on all 90 machines in your webcluster turns into a very tedious job when done by hand. There are several tools that are available to ease this, including puppet or cfengine. For cases where those tools are overkill, various solutions based on rsync and subversion (or another VCS) are available. Along with this comes the problem of asset management, keeping track of which machines you have, as well as where they are and what they are doing can be eased with tools like rackmonkey.

User Authentication

A new user in your organization starts out as easy to deal with, but as the number of users and services grows, adding or removing user accounts becomes non-trivial. In an all UNIX shop, LDAP is an attracive choice for putting data about users in one place and accessing it from many services, whether it be apache, sshd, or mail clients looking to build address books. Older technologies include the veneriable NIS, which is still an option depending on your needs, and in a mixed windows/unix environment, Active Directory becomes an attractive choice.

I hope that you've found some google fodder, we'll be looking at these items closer in future articles. There is a world of tools out there, and of course I can't hope to cover them all. Deploying any of these tools or techniques has a cost, and whether paying that cost is the right thing to do or not is always a judgement call. You'll never get any payback from an LDAP infrastructure deployment if you never grow beyond one machine. On the other hand, if you have five hundred machines and no LDAP you've potentially wasted uncountable hours managing individual machines. As Yogi Berra once said, *Predictions are hard, especially about the future*. If your predictions are wrong you'll either spend time deploying a solution to a problem you'll never have, or kicking yourself for not deploying the solution that will make your life easier as you add a user to your 77th webserver.

JOSH PAETZEL

A 37 year old advocate, user and developer of BSD UNIX based systems. he resides in Minneapolis, Minnesota, USA where he hacks on FreeBSD and PC-BSD, both as a volunteer and as part of his full time work as the Director of IT at iXsystems.

Testing Hammer Deduplication on Real-world Data

If you've been in the market for storage devices lately, you may have noticed a trend. Prices for various storage devices are generally determined by size, then speed, and then the whole price is increased by the features that come with that disk appliance. The ability to reduce disk usage by recording disk structures once and then just leaving references to that structure any place it is repeated is one of those features. *Deduplication* is the name, and you can count on it adding an extra zero on the price.

Hammer is DragonFly BSD's default file system. It allows infinite snapshots, quick recovery, historical snapshots, and master/slave cloning of individual pseudo-filesystems, among other things. It also allows deduplication, which can be run as part of the batch processes for cleaning up a Hammer volume.

Hammer looks at disk data at the block level, using the CRC32 checksum for each block to determine if the block is a candidate for deduplication. Hammer already constructs CRC32 checksums for every block, so this reduces the work to a simple lookup to find all possible deduplication candidates. However, since CRC32 checksums are not a guarantee of an exact match, the actual data is then compared in these blocks.

If there's an exact match for every byte in the blocks being compared, the duplicate block is removed and a reference made to its former location, pointing back to the remaining, matching block.

How much space does this save? It depends, of course. If there's a lot of repeated data on a disk, like multiple backup files of the same data, it can make a big

Listing 1. *Initial disk volumes*

```
# df
Filesystem                     1K-blocks      Used     Avail Capacity  Mounted on
ROOT                           966000640 566434576 399566064    59%    /
devfs                                  1         1         0   100%    /dev
/dev/serno/WD-WCAU49270429.s1a    257998    142248     95112    60%    /boot
/pfs/@@-1:00001                966000640 566434576 399566064    59%    /var
/pfs/@@-1:00002                966000640 566434576 399566064    59%    /tmp
/pfs/@@-1:00003                966000640 566434576 399566064    59%    /usr
/pfs/@@-1:00004                966000640 566434576 399566064    59%    /home
/pfs/@@-1:00005                966000640 566434576 399566064    59%    /usr/obj
/pfs/@@-1:00006                966000640 566434576 399566064    59%    /var/crash
/pfs/@@-1:00007                966000640 566434576 399566064    59%    /var/tmp
procfs                                 4         4         0   100%    /proc
```

difference. It'll have less of an impact on rapidly-changing data, or very heterogenous material.

Real-world tests

All this sounds great, but it's necessary to see what happens when reality crashes into the picture. I explored this using my own DragonFly host. The system I tested this on is running with a 1 terabyte drive, which was 59% full when I started my testing (Listing 1).

Each of the lines starting with `/pfs/...` are pseudo-file-systems, or PFS, all located within the root Hammer volume. Note that since they are all part of the same volume, they report the same disk usage. A PFS is used to segregate Hammer history and snapshot policies.

You may notice that `/boot` is a separate volume, formatted as UFS, and mounted by serial number (another DragonFly feature). It's possible to have /boot be a HAMMER volume, but I didn't do that with this installation.

Predicting the savings

It's possible to figure out a rough ratio of disk savings from deduplication with the `dedup-simulate` command. It varies depending on the contents of each PFS. For example, here's the results for `/var`: see Listing 2.

And here's the results for `/usr`: see Listing 3.

So it looks like I'd save somewhere between 10 and 30 percent of disk space, if you averaged out the results. (I didn't reprint the whole list here; just the two extremes.)

Listing 2. *Deduplicating /var*

```
# hammer dedup-simulate /var
Dedup-simulate /var: objspace 8000000000000000:0000
                    7fffffffffffffff:ffff pfs_id 1
Dedup-simulate /var succeeded
Simulated dedup ratio = 1.10
```

Listing 3. *Deduplicating /usr*

```
# hammer dedup-simulate /usr
Dedup-simulate /usr: objspace 8000000000000000:0000
                    7fffffffffffffff:ffff pfs_id 3
Dedup-simulate /usr succeeded
Simulated dedup ratio = 1.50
```

Listing 4. *Version Errors*

```
# hammer dedup /home
Dedup /home: objspace 8000000000000000:0000
                    7fffffffffffffff:ffff pfs_id 4
HAMMER filesystem must be at least version 5 to dedup
```

Version support and upgrades

I ran into a snag when I started the deduplication process, however. This system had been running earlier versions of Hammer which did not support deduplication (Listing 4).

Upgrading a Hammer filesystem is easy, though. It can be upgraded in-place, and takes very little time. Upgrading the 1T disk holding this data took less than a minute.

```
# hammer version-upgrade /home 5
            hammer version-upgrade:
              succeeded
```

Listing 5. *Deduplication output*

```
# hammer dedup /usr
Dedup /usr: objspace 8000000000000000:0000
                    7fffffffffffffff:ffff pfs_id 3
Dedup /usr succeeded
Dedup ratio = 1.49
       63 GB referenced
       42 GB allocated
       22 MB skipped
        392 CRC collisions
          0 SHA collisions
          0 bigblock underflows
# hammer dedup /var
Dedup /var: objspace 8000000000000000:0000
                    7fffffffffffffff:ffff pfs_id 1
Dedup /var succeeded
Dedup ratio = 1.10

     2580 MB allocated
       14 MB skipped
          1 CRC collisions
          0 SHA collisions
          0 bigblock underflows
```

Listing 6. *The almost-final results*

```
# df
Filesystem                       1K-blocks      Used     Avail Capacity  Mounted on
ROOT                             966000640 505887504 460113136    52%    /
devfs                                    1         1         0   100%    /dev
/dev/serno/WD-WCAU49270429.s1a      257998    142248     95112    60%    /boot
/pfs/@@-1:00001                  966000640 505887504 460113136    52%    /var
/pfs/@@-1:00002                  966000640 505887504 460113136    52%    /tmp
/pfs/@@-1:00003                  966000640 505887504 460113136    52%    /usr
/pfs/@@-1:00004                  966000640 505887504 460113136    52%    /home
/pfs/@@-1:00005                  966000640 505887504 460113136    52%    /usr/obj
/pfs/@@-1:00006                  966000640 505887504 460113136    52%    /var/crash
/pfs/@@-1:00007                  966000640 505887504 460113136    52%    /var/tmp
procfs                                   4         4         0   100%    /proc
```

And that's it! Onward to disk savings! I'm just reprinting results from two PFS deduplications here (Listing 5).

So what were the savings? A bit less than 15% of existing data: see Listing 6.

It seemed a bit on the low end, but given that it was 'free', I wasn't going to argue. An interesting thing happened with the overnight cleanup of Hammer, which made me take a second look. Usage dropped another 2% once the disk was able to go through reblocking and reorganizing the data. In fact, fully reblocking the disk so that data was better organized (and therefore easier to find matches in data arrangement) brought the disk usage levels down to 49 percent.

The answer here seems to be in reblocking. That increase in available space seems to imply I wasn't allowing the disk to finish its reblocking cycle completely. Time and frequency for all the Hammer activities are controlled with the *hammer viconfig* command. Reblocking defaults to only 5 minutes every day, so if you have a large or frequently changing volume, it's worth increasing it.

This system was also a fresh upgrade to a version of Hammer that supported deduplication, so it's possible the reblocking wouldn't be necessary in a newly created volume or a volume that had been running at that Hammer version level for some time. In either case, increasing the run time for the various Hammer cleanup activities won't generally have a negative effect if the activities complete before the end of the allotted time.

TL;DR Summary

With less than an hour's work, and without having to take any disks offline, I recovered about 10% of my total disk space, at no apparent cost.

Cost per gigabyte of drive space is so very cheap these days that it translates to a cost savings of perhaps USD$10 for my 1TB disk. Still, it's a *free* benefit, compared to the much higher costs of the specialized storage systems that offer deduplication now as commercial products.

Also, if you have no additional disk space, this is a near-instant way to increase disk space without touching hardware. In certain situations, it could be a life saver.

JUSTIN C. SHERRILL

Justin Sherrill has been publishing the DragonFly BSD Digest since 2004, and is responsible for several other parts of DragonFly that aren't made out of code. He lives in the northeast United States and works over a thousand feet underground.

PC-BSD's New Control Panel

This article introduces the new Control Panel that will ship with PC-BSD 9.0. Readers are encouraged to try out the Control Panel prior to release by downloading a PC-BSD 9.0 testing snapshot or building the Control Panel on a PC-BSD 8.x system or a FreeBSD 8.x system that has Xorg configured.

PC-BSD 9.0, due for release later this year, will introduce a graphical Control Panel for the administration of a BSD system. Prior releases of PC-BSD added some BSD-specific configuration utilities to those already provided by the KDE System Settings menu. Figure 1 shows a screenshot of System Settings from a PC-BSD 8.2 system.

While System Settings contains most of the utilities needed to configure a system, it does not distinguish between which utilities came with KDE and which were custom created for PC-BSD. For example, under the Network and Connectivity section shown in Figure 1, Network Settings and Sharing are KDE utilities. The Firewall utility was created by the PC-BSD project to provide graphical access to the PF firewall and the System Network Configuration utility was created to understand FreeBSD network device names, `wpa_supplicant`, `ppp.conf`, and `rc.conf`.

One of the design features of PC-BSD 9.0 is the ability for users to install multiple desktop environments both during and after installation of the operating system. Users can choose from the following desktop environments: KDE4, GNOME2, XFCE4, LXDE, Awesome, Enlightenment, IceWM, ScrotWM, and Window Maker. Since each desktop environment installs its own set of configuration utilities, there is a clear need for a unified Control Panel. A Control Panel makes it easy for users to find and use a consistent set of tools to configure their system, regardless of which desktop environment they are currently logged into.

Control Panel Features

Kris Moore created the initial design of the Control Panel and Yuri Momotiuk extended the design to include support for localizations and the ability to include the configuration

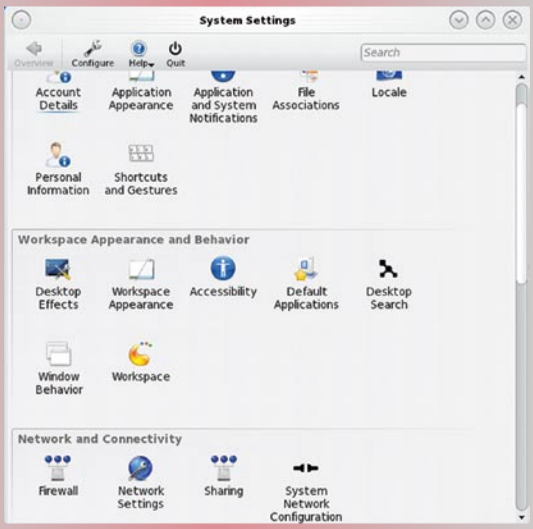


Figure 1. *System Settings Menu*

applications from other installed desktop environments. Control Panel's features include:

- a graphical design based on the QT toolkit.
- XDG-compliance. XDG is an interoperability standard for desktop environments that run on top of the X window system (see *http://freedesktop.org* for more details). From a user point of view, this means that Control Panel will be automatically integrated into the menu of any XDG-compliant desktop and a shortcut can be added to that desktop or its panel.
- support for localization. This means that the text of the Control Panel's icons as well as the menu text for each PC-BSD configuration application will be translated. You can see the status of the translation for each supported language at the PC-BSD Pootle site (*http://pootle.pcbsd.org*). If you are interested in assisting with a translation, instructions to get you started are available at *http://wiki.pcbsd.org/index.php/Become_a_Translator.*
- support for locale settings and the keyboard layouts and variants that can be selected during the installation of PC-BSD (*http://wiki.pcbsd.org/index.php/Keyboard_Selection_Screen*).
- a desktop selector menu which is described in more detail in the section *Switching Control Panel Desktop Entries*. The back-end for this functionality is a de-info shell script which is used to determine which XDG-compliant desktops are installed. It currently supports the KDE3, KDE4, GNOME2, XFCE3, XFCE4, and LXDE desktop environments. The desktop selector menu only appears when a minimum of two supported desktops are installed.

If you would like to try out Control Panel, there are two ways to install it:

- Download and install a PC-BSD 9.0 testing snapshot from *ftp://ftp.pcbsd.org/pub/snapshots/.* Control Panel is built into PC-BSD 9.0 and will appear as an icon on the desktop if you select KDE, GNOME, XFCE, or LXDE during the installation. For testing purposes, PC-BSD 9.0 can be installed into a virtual environment such as VirtualBox.
- Download and compile the source for Control Panel. The instructions in the next section have been tested on FreeBSD 8.2 and PC-BSD 8.2.

Installing Control Panel on FreeBSD or PC-BSD 8.2

If you are installing Control Panel on a PC-BSD 8.2 system, you already have all of the required dependencies. You will also need to have src installed; if it is not, you can install it using *System Settings->System Manager->Tasks->Fetch System Source*. If you would like to try out the desktop selector menu, install at least one of the following desktops: GNOME2, XFCE3, XFCE4, or LXDE using Software Manager or `pkg_add -r`.

Depending upon what you already have installed on your FreeBSD system, you may have to add some packages first. Use `pkg_info` to see if the following packages are already installed and `pkg_add -r` to install any missing packages:

- `qt4-qmake`
- `qt4-linguist`
- `qt4-uic`
- `qt4-moc`
- `qt4-rcc`

A FreeBSD system will also need to have the following installed and configured:

- `xorg`
- `/usr/src/`
- at least one desktop environment. If you wish to try out the desktop selector menu, install at least 2 of the following desktops: KDE3, KDE4, GNOME2, XFCE3, XFCE4, or LXDE.

To download and compile Control Panel, issue the following commands as the superuser. The commands which are highlighted in yellow are only required on a FreeBSD system. Whatever directory you are in when you issue the `svn` command is where the downloaded source will be stored. The download size is about 40 MB.

```
svn co svn://svn.pcbsd.org/pcbsd/current/src-qt4
cd src-qt4
qmake-qt4 *.pro
cd libpcbsd
qmake-qt4 *.pro
make install
cd ..
make && make install
```

To also install the desktop selector menu, run the following commands:

```
cd ..
svn co svn://svn.pcbsd.org/pcbsd/current/src-sh/de-info
cd de-info
make install
```

If your current desktop environment is KDE, an entry for Control Panel will be automatically added to *Applications ->System->PC-BSD Control Panel*.

If your current desktop environment is GNOME, an entry for Control Panel will be automatically added to *System-> Administration->PC-BSD Control Panel*.

If your current desktop environment is XFCE, an entry for Control Panel should be automatically added to *System->PC-BSD Control Panel*.

If your current desktop environment is LXDE, an entry for Control Panel should be automatically added to *System->PC-BSD Control Panel*.

You can also start Control Panel by typing `pc-controlpanel`. However, you won't have superuser access for the configuration utilities unless you start that command with a switch-user command such as `sudo`, `kdesu`, or `gksu`.

Switching Control Panel Desktop Entries

Figure 2 shows a screenshot of Control Panel with the desktop selector menu open.

NOTE

The desktop selector menu will not appear on an 8.2 system if you did not install de-info or if only one supported desktop is installed.

In the example shown in Figure 2, the user is currently logged into the LXDE desktop on a system that has KDE, GNOME, XFCE, and LXDE installed. The user has selected to add the GNOME configuration tools to their Control Panel–you can tell that this is the case as the icon for the desktop selector menu has changed to the GNOME foot. By making this selection, several utilities have been integrated into the Control Panel's sections and a new section called *Desktop environment (Gnome)* has been added to Control Panel. In other words, the user can now access the GNOME configuration utilities while logged into the LXDE desktop. The user also has the option to add the KDE configuration utilities (which will remove the GNOME ones), to add the XFCE utilities, or to add all of the configuration utilities from all of the installed desktops.

If an entry for a supported desktop does not appear in the desktop selector menu, it means that that desktop is not currently installed.

On a FreeBSD system, you can install additional desktops using packages or ports.

On a PC-BSD 8.2 system, you can install additional desktops using Software Manager.

On a PC-BSD 9.0 system, you can install additional desktops using Control *Panel->System Manager->System Packages*. Simply select the desktop(s) you wish to install from the menu shown in Figure 3.

Default Control Panel Icons

Regardless of the desktop in use, the following icons are specific to PC-BSD and are always available from Control Panel:

AppCafe

Is the new software management tool in PC-BSD 9.0. PC-BSD uses the PBI (push button installer) format for managing software. The PBI format has been redesigned for 9.0, adding new functionality such as a hashdir to manage dependencies, incremental upgrades, the ability to create custom software repositories, and the ability for users to install desktop software without needing superuser access. AppCafe provides a graphical utility for finding PBIs and software repositories, managing software installation and removal, upgrading installed software, and creating desktop icons for installed software. When software is installed using AppCafe, even novice users can safely install and uninstall PBIs without inadvertently overwriting or deleting

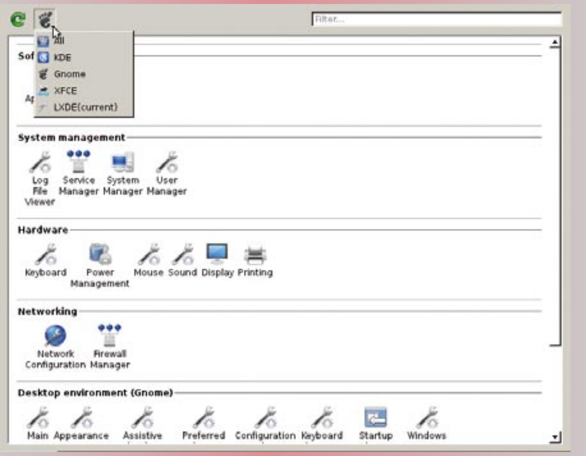
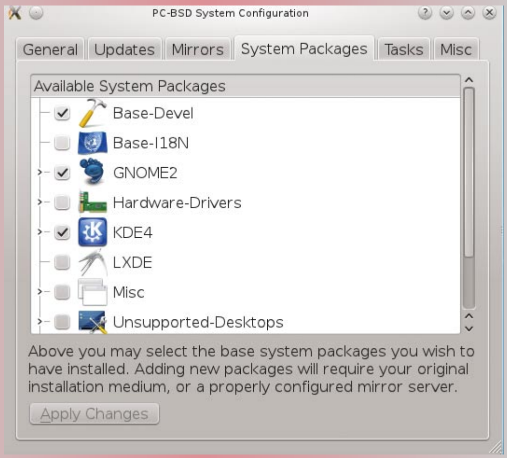


Figure 2. *Using the Desktop Selector Menu*



Above you may select the base system packages you wish to have installed. Adding new packages will require your original installation medium, or a properly configured mirror server.

Figure 3. *System Packages Menu of System Managera*

files needed by the operating system or other applications. When a regular user installs an application, they have the option to *add to desktop for all users*, meaning that an application only needs to be installed once on a multi-user system. AppCafe is also localized, meaning that all translated software descriptions will appear in the selected language. More information about using AppCafe can be found at *http://wiki.pcbsd.org/index.php/Using_AppCafe*.

Ports Jail

Provides a command line environment where users who are new to FreeBSD packages and ports can safely experiment and learn how to use the FreeBSD software management command line tools without affecting the software that was installed with the operating system. More information on using Ports Jail can be found at *http://wiki.pcbsd.org/index.php/Using_Ports_Jail_to_Manage_FreeBSD_Packages_and_Ports*.

Service Manager

Provides an easy-to-use graphical utility for managing PC-BSD services; in other words, it is a front-end to rc.conf. The listed services can be enabled/disabled at system startup and their current status can be toggled between stopped and running. More information on Service Manager can be found at *http://wiki.pcbsd.org/index.php/Service_Manager*.

System Manager

Can be used to create a diagnostic report (e.g. the outputs of `dmesg`, `/var/log/messages`, `top`, and `pciconf -lv`) which you can send to a mailing list to assist when troubleshooting your system. This utility is also used to apply system updates and security patches, specify an update mirror, install or uninstall components such as window managers, drivers, and MythTV, install ports and src, and set the boot splash image. More information on System Manager is available from *http://wiki.pcbsd.org/index.php/System_Manager*.

User Manager

Allows you to easily add and delete users and groups on your system, as well as change a user's or the administrative password. More information on User Manager is available from *http://wiki.pcbsd.org/index.php/User_Manager*.

Display

Used to change the display driver or video resolution settings. More information about Display Manager is available from *http://wiki.pcbsd.org/index.php/Display*.

Printing

Provides a front-end to CUPS allowing you to add and remove printers and manage your printer settings. More information about the Printing Manager is available from *http://wiki.pcbsd.org/index.php/Printing*.

Network Configuration

Allows you to view or manually configure Ethernet and wireless network devices, 3G and PPP settings, and proxy configuration. More information about Network Configuration is available from *http://wiki.pcbsd.org/index.php/System_Network_Configuration*.

Firewall Manager

Provides a graphical front-end to the PF firewall. It allows you to stop, start, or restart the firewall, view existing firewall rules, and add/remove firewall rules. More information about Firewall Manager is available from *http://wiki.pcbsd.org/index.php/Firewall_Manager*.

Life Preserver

Allows you to schedule backups of your PC-BSD system to a remote system using rsync and SSH. The August issue of BSD Mag will have an article describing how to use Life Preserver. More information on Life Preserver is also available from *http://wiki.pcbsd.org/index.php/Life_Preserver*.

Summary

The Control Panel in PC-BSD 9.0 makes it easy for users to access the graphical configuration utilities that they need, regardless of which desktop environment they are currently logged into.

Readers are encouraged to try out the Control Panel and to report any bugs or feature requests to the PC-BSD testing mailing list: *http://lists.pcbsd.org/mailman/listinfo/testing*.

DRU LAVIGNE

Dru Lavigne is author of BSD Hacks, The Best of FreeBSD Basics, and The Definitive Guide to PC-BSD. As Director of Community Development for the PC-BSD Project, she leads the documentation team, assists new users, helps to find and fix bugs, and reaches out to the community to discover their needs. She is the former Managing Editor of the Open Source Business Resource, a free monthly publication covering open source and the commercialization of open source assets. She is founder and current Chair of the BSD Certification Group Inc., a non-profit organization with a mission to create the standard for certifying BSD system administrators, and serves on the Board of the FreeBSD Foundation.

Using POSTGIS tabular

and geographic data with FreeBSD

In this article, we will look at extending our GIS server to use PostGIS.

What you will learn…

- How to use spatial data with Geoserver and FreeBSD

What you should know…

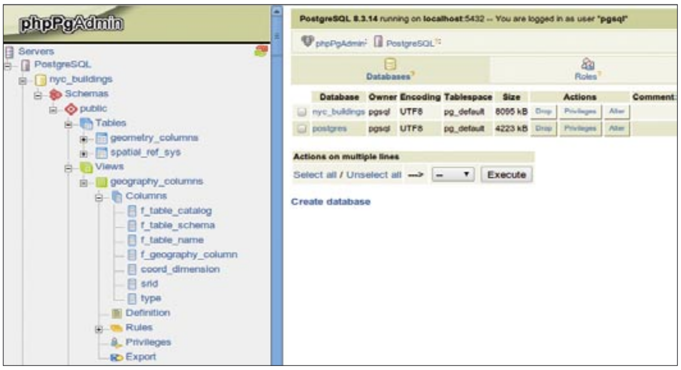
- BSD administration skills, how to install applications from source

In the previous article, we configured Geoserver, PostgreSQL and PostGIS and loaded some test data into Geoserver for New York City. This gives us the basic *graphical* representation of our map. However, while it is possible to embed points or outlines in a graphical format and present this as layers (e.g. TIFF) on a large scale, this process is cumbersome, inefficient and does not allow the operator to easily calculate features such as an area. For instance, if we wanted to plot the distribution of car ownership by geographical region, we could use a desktop tool to manipulate a layer, plot the points and boundaries etc. and add the statistical data to a flat file. If at a later date we wanted to analyse the data more rigorously e.g. selecting an area where car ownership is prevalent, we would be limited in what we could achieve. This is where a dedicated spatial database comes to our rescue, geographical features such as boundaries, points etc. can be stored efficiently in a database, and other discrete features such as population, elevation etc. can be queried and cross-referenced. The best of both worlds can then be achieved, static or *slow changing* data such as the aerial graphical representation of the map can be cached and served as an image, and the variable or *fast changing* data such as population, income etc. can be manipulated and displayed via a database back end. This does not negate the requirement of using a desktop tool to initially plot the dataset, but it does open the door to allow the efficient interpretation and analysis of statistical data.

Figure 1. *The phpPgAdmin web interface*

Why PostgreSQL and PostGIS?

While various databases offer rudimentary storage capabilities, PostgreSQL with PostGIS extensions is considered the most mature Open Source candidate offering geometric functions such as distance, area, union, intersection etc. and other speciality geometry data types. In other words, PostGIS spatially enables and extends a



PostgreSQL 8.3.14 running on localhost:5432 -- You are logged in as user "pgsql"

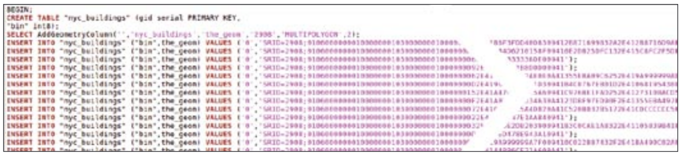
	Database	Owner	Encoding	Tablespace	Size	Actions			Comment
□	nyc_buildings	pgsql	UTF8	pg_default	8095 kB	Drop	Privileges	Alter	
□	postgres	pgsql	UTF8	pg_default	4223 kB	Drop	Privileges	Alter	

Actions on multiple lines

Figure 2. *nyc_buildings table with spatial extensions* relatively standard database to store geographic data (e.g. shapes of features) in tabular format. PostgreSQL *out of the box* can handle native geometry data, but this functionality is not mature enough for more complex geometry types. For further details on PostGIS, please see the Refractions Research website at *http://postgis.refractions.net*. When initially investigating candidates for the GIS project, the author considered using MySQL, but it rapidly became apparent that the GIS community has extensive support for PostGIS so the decision was made to stick to proven technology.

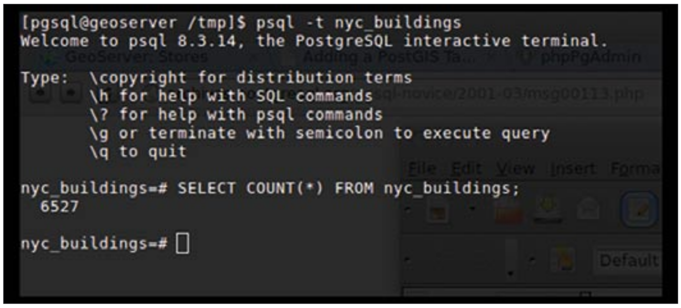
Getting under the hood

There is nothing like an SQL query to illustrate complex datasets, and although PostgreSQL has a powerful CLI, other users may prefer to use a browser-based or graphical interface under Xorg to access and manage the database. If the web route is chosen, a server or workstation will need to be configured with PHP5, PHP Extensions (for the pgsql extension), Apache, and phpPgAdmin. This will require building PHP from source as the Apache module needs to be specifically enabled. PhpPgAdmin will also require the `config.inc.php` file to be modified to allow the pgsql user to initially login (Change the `$conf['extra_login_security']` from



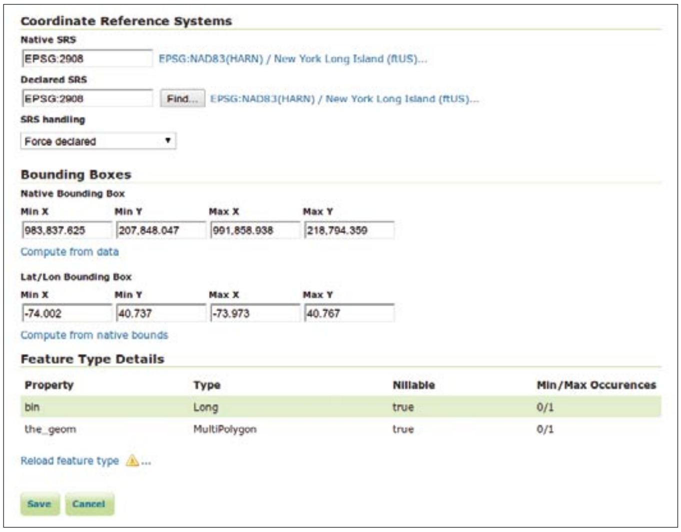
```
BEGIN;
CREATE TABLE "nyc_buildings" (gid serial PRIMARY KEY,
"bin" int8);
SELECT AddGeometryColumn('','nyc_buildings','the_geom','2908','MULTIPOLYGON',2);
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]');
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]');
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]');
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]');
INSERT INTO "nyc_buildings" ("bin",the_geom) VALUES ('0','SRID=2908;[illegible]
```

Figure 3. *Truncated nyc_buildings.sql with geometry data*



```
[pgsql@geoserver /tmp]$ psql -t nyc_buildings
Welcome to psql 8.3.14, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help with psql commands
       \g or terminate with semicolon to execute query
       \q to quit

nyc_buildings=# SELECT COUNT(*) FROM nyc_buildings;
  6527

nyc_buildings=# 
```

Figure 4. *PostgreSQL Interactive Terminal*



Coordinate Reference Systems

Native SRS

EPSG:2908 EPSG:NAD83(HARN) / New York Long Island (ftUS)...

Declared SRS

EPSG:2908 Find... EPSG:NAD83(HARN) / New York Long Island (ftUS)...

SRS handling

Force declared ▾

Bounding Boxes

Native Bounding Box

Min X	Min Y	Max X	Max Y
983,837.625	207,848.047	991,858.938	218,794.359

Compute from data

Lat/Lon Bounding Box

Min X	Min Y	Max X	Max Y
-74.002	40.737	-73.973	40.767

Compute from native bounds

Feature Type Details

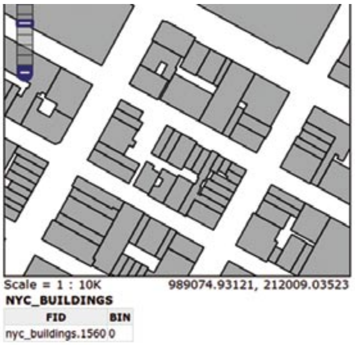
Property	Type	Nillable	Min/Max Occurences
bin	Long	true	0/1
the_geom	MultiPolygon	true	0/1

Reload feature type ⚠...

Figure 5. *Computing the Bounding Box*

Table 1. *nyc datastore connection parameters*

Data Source	nyc_buildings
Decription	nyc buildings
Host	localhost
Port	5432
Database	nyc_buildings
schema	public
User	pgsql
Password	pgsql

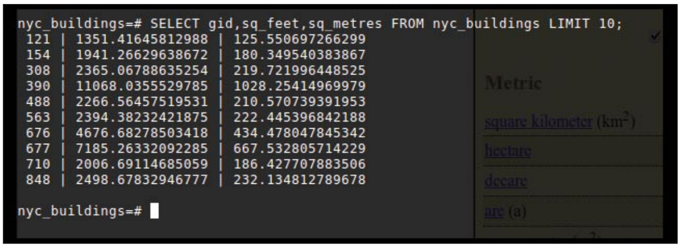


Scale = 1 : 10K 989074.93121, 212009.03523

NYC_BUILDINGS

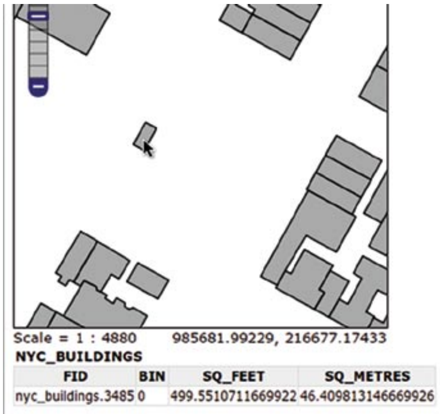
FID	BIN
nyc_buildings.1560	0

Figure 6. *nyc_building data served from the PostgreSQL database (Zoomed in)*



```
nyc_buildings=# SELECT gid,sq_feet,sq_metres FROM nyc_buildings LIMIT 10;
 121 | 1351.41645812988 | 125.550697266299
 154 | 1941.26629638672 | 180.349540383867
 308 | 2365.06788635254 | 219.721996448525
 390 | 11068.0355529785 | 1028.25414969979
 488 | 2266.56457519531 | 210.570739391953
 563 | 2394.38232421875 | 222.445396842188
 676 | 4676.68278503418 | 434.478047845342
 677 | 7185.26332092285 | 667.532805714229
 710 | 2006.69114685059 | 186.427707883506
 848 | 2498.67832946777 | 232.134812789678

nyc_buildings=# 
```

Figure 7. *Building coverage in sq feet and metres*



Scale = 1 : 4880 985681.99229, 216677.17433

NYC_BUILDINGS

FID	BIN	SQ_FEET	SQ_METRES
nyc_buildings.3485	0	499.5510711669922	46.40981314669926

Figure 8. *Openlayers map showing area*

Listing 1. *Attribute based polygon SLD*

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<StyledLayerDescriptor version="1.0.0"
xsi:schemaLocation="http://www.opengis.net/sld StyledLay
                    erDescriptor.xsd"
xmlns="http://www.opengis.net/sld"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <NamedLayer>
    <Name>Attribute-based polygon</Name>
    <UserStyle>
      <Title>Real Estate Coverage</Title>
      <FeatureTypeStyle>

        <Rule>
          <Name>Band1</Name>
          <Title>Less Than 1000 sq feet</Title>
          <ogc:Filter>
            <ogc:PropertyIsLessThan>
              <ogc:PropertyName>sq_feet</ogc:
                    PropertyName>
              <ogc:Literal>1000</ogc:Literal>
            </ogc:PropertyIsLessThan>
          </ogc:Filter>
          <PolygonSymbolizer>
            <Fill>
              <CssParameter name="fill">#B7C0EA</
                    CssParameter>
            </Fill>
            <Stroke>
              <CssParameter name="stroke">#DADADA</
                    CssParameter>
              <CssParameter name="stroke-width">1</
                    CssParameter>
            </Stroke>
          </PolygonSymbolizer>
        </Rule>

        <Rule>
          <Name>Band2</Name>
          <Title>1,000 to 3,000 sq feet</Title>
          <ogc:Filter>
            <ogc:And>
              <ogc:PropertyIsGreaterThanOrEqualTo>
                <ogc:PropertyName>sq_feet</ogc:
                    PropertyName>
                <ogc:Literal>1000</ogc:Literal>
              </ogc:PropertyIsGreaterThanOrEqualTo>
              <ogc:PropertyIsLessThan>
                <ogc:PropertyName>sq_feet</ogc:
                    PropertyName>
                <ogc:Literal>3000</ogc:Literal>
              </ogc:PropertyIsLessThan>
            </ogc:And>
          </ogc:Filter>
          <PolygonSymbolizer>
            <Fill>
             <CssParameter name="fill">#B26585</CssParameter>
            </Fill>
            <Stroke>
             <CssParameter name="stroke">#DADADA</CssParameter>
              <CssParameter name="stroke-width">1</
                    CssParameter>
            </Stroke>
          </PolygonSymbolizer>


        <!-- Repeat the code between <Rule></Rule> for
                    Band3 - Band6
             using ogc:Literal and CssParameter
                    name="fill" values as appropriate -->

        <Rule>
          <Name>Band7</Name>
          <Title>Greater Than 100,000 sq feet</Title>
          <ogc:Filter>
            <ogc:PropertyIsGreaterThan>
              <ogc:PropertyName>sq_feet</ogc:PropertyName>
              <ogc:Literal>100000</ogc:Literal>
            </ogc:PropertyIsGreaterThan>
          </ogc:Filter>
          <PolygonSymbolizer>
            <Fill>
             <CssParameter name="fill">#FF66B9</CssParameter>
            </Fill>
            <Stroke>
              <CssParameter name="stroke">#DADADA</
                    CssParameter>
             <CssParameter name="stroke-width">1</CssParameter>
            </Stroke>
          </PolygonSymbolizer>
        </Rule>

      </FeatureTypeStyle>
    </UserStyle>
  </NamedLayer>
</StyledLayerDescriptor>
```

true->false) but this should not be used in production – it is preferable to create another database user with sufficient rights. The `$conf['servers']` setting will also need to be configured to point to the PostgreSQL host. Webmin could also be used, as it has basic PostgreSQL support, but it has limited graphical capabilities for showing the database structure. See (Figure 1). For most operations, you can login to the server using SSH and run the PostgreSQL client utility. Alternatively, PostgreSQL could be configured to allow client / server access from another workstation. For this tutorial, I will use a combination of SSH access with phpPgAdmin screenshots to illustrate the database structure and contents.

Getting started

In the last howto, we installed PostgreSQL and PostGIS, but first we need to create the database before we can create the Geoserver store. Set the pgsql password (I am using pgsql for the demo but this is obviously not secure), create a database and add the PostGIS extensions to the table using the SQL query provided by PostGIS. After SSH'ing into our Geoserver box su to root and perform the following:

```
 passwd pgsql
 su pgsql
 createdb nyc_buildings
 createlang plpgsql nyc_buildings
 psql -d nyc_buildings -f /usr/local/share/postgis/contrib/
postgis-1.5/postgis.sql
 psql -d nyc_buildings -f /usr/local/share/postgis/contrib/
postgis-1.5/spatial_ref_sys.sql
```

This will create a spatially aware database nyc_buildings – see (Figure 2). We now need to download some sample data for our PostgreSQL store from the Geoserver website and import:

```
cd /tmp
wget http://docs.geoserver.org/stable/en/user/_downloads/
                    nyc_buildings.zip
unzip nyc_buildings.zip
```

If you are to view the SQL file with view or cat you should see geometry data similar to (Figure 3). We can now import this into our newly created table `nyc _ buildings`:

```
psql -f nyc_buildings.sql nyc_buildings
```

Check that the data has imported OK. Enter the PostgreSQL interactive terminal and perform a count (Figure 4):

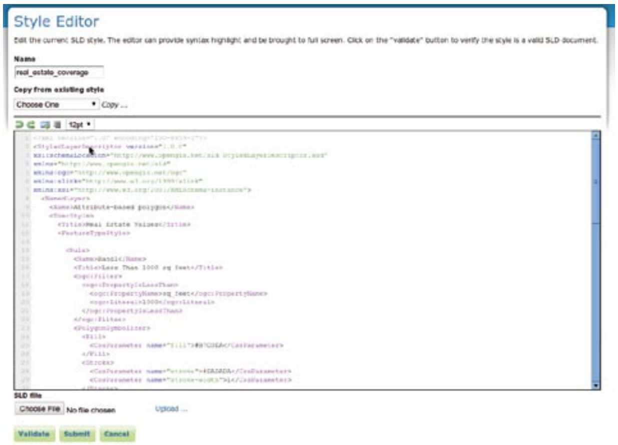
```
psql -t nyc_buildings
SELECT COUNT(*) FROM nyc_buildings;
\q
```

Login to Geoserver (username: admin/password: geoserver) and create a new PostGIS data store in the tiger workspace with the following criteria and then save and publish (Table 1).

When you click on the publish link, you will be prompted for the Native and Lat/Long Bounding Boxes co-ordinates. Click on Compute from data and Compute from native bounds. Click on the Save button to create the new layer (Figure 5). Click on Layer Preview and then the link for Openlayers for the `tiger:nyc_buildings` layer. You should now be presented with the nyc_buildings data served from the PostGIS table (Figure 6).

Manipulating our data

So far, so good – we have a grey map of all the buildings in Manhattan. It would be useful to view the map and display the floor size of each building in square feet and metres



Style Editor

Edit the current SLD style. The editor can provide syntax highlight and be brought to full screen. Click on the "validate" button to verify the style is a valid SLD document.

Name: real_estate_coverage

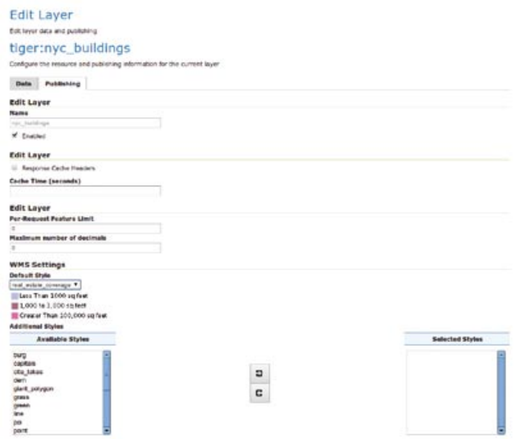
Copy from existing style: Choose One ▾ Copy ...

12pt ▾

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<StyledLayerDescriptor version="1.0.0"
xsi:schemaLocation="http://www.opengis.net/sld StyledLayerDescriptor.xsd"
xmlns="http://www.opengis.net/sld"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <NamedLayer>
    <Name>Attribute-based polygon</Name>
    <UserStyle>
      <Title>Real Estate Values</Title>
      <FeatureTypeStyle>

        <Rule>
          <Name>Band1</Name>
          <Title>Less Than 1000 sq feet</Title>
          <ogc:Filter>
            <ogc:PropertyIsLessThan>
              <ogc:PropertyName>sq_feet</ogc:PropertyName>
              <ogc:Literal>1000</ogc:Literal>
            </ogc:PropertyIsLessThan>
          </ogc:Filter>
          <PolygonSymbolizer>
            <Fill>
              <CssParameter name="fill">#B7C0EA</CssParameter>
            </Fill>
            <Stroke>
              <CssParameter name="stroke">#DADADA</CssParameter>
              <CssParameter name="stroke-width">1</CssParameter>
```

SLD file: Choose File No file chosen Upload ...

Figure 9. *Style editor with SLD loaded*



Edit Layer

Edit layer data and publishing

tiger:nyc_buildings

Configure the resource and publishing information for the current layer

Edit Layer

Name: nyc_buildings

☑ Enabled

Edit Layer

□ Response Cache Headers

Cache Time (seconds):

Edit Layer

Per-Request Feature Limit: 0

Maximum number of decimals: 0

WMS Settings

Default Style: real_estate_coverage ▾

- Less Than 1000 sq feet
- 1,000 to 1,000 sq feet
- Greater Than 100,000 sq feet

Additional Styles

Available Styles	Selected Styles
burg capitals cite_lakes dem giant_polygon grass green line poi point	

Figure 10. *nyc_buildings layer with real_estate_coverage default style*

Table 2. *SLD settings for Figure 12*

Name	Title	OGC:			
		PropertyIsGreaterThanOrEqualTo	PropertyIsLessThan	PropertyIsLessThan	CssParameter name="fill"
		Literal	Literal	Literal	
Band1	Less Than 1000 sq feet		1000		#B7C0EA
Band2	1,000 to 3,000 sq feet	1000	3000		#B26585
Band3	3,000 to 5,000 sq feet	3000	5000		#F29A28
Band4	5,000 to 10,000 sq feet	5000	10000		#E5DCB5
Band5	10,000 to 50,000 sq feet	10000	50000		#BAB2D9
Band6	50,000 to 100,000 sq feet	50000	100000		#F29A98
Band7	Greater Than 100,000 sq feet			100000	#FF66B9

and in a different colour depending on the surface area each building occupies. This is achieved in two steps, first we will need to extract the area of each building (which is a polygon), calculate the area using the PostGIS function ST_Area, and add this to our table. Once we have our additional data set, we can construct an *XML Styled Layer Descriptor* (SLD) which will render the area according to our definition.

SSH into the server, su to pgsql and enter the PostgreSQL terminal:

```
su pgsql
psql -t nyc_buildings
ALTER TABLE nyc_buildings ADD COLUMN sq_feet double precision;
ALTER TABLE nyc_buildings ADD COLUMN sq_metres double
                    precision;
UPDATE nyc_buildings SET sq_feet = ST_Area(the_geom);
UPDATE nyc_buildings SET sq_metres = ST_Area(the_geom)
                    *POWER(0.3048,2);
SELECT gid,sq_feet,sq_metres FROM nyc_buildings LIMIT 10;
\q
```

The SELECT statement should return 10 records with the building area in square feet and metres (Figure 7).

Return to the `nyc_building` layer in Geoserver, and click on the link reload feature type. This will force Geoserver to publish the new parameters. On browsing the map in Layer preview, clicking on a building will display the polygon coverage (Figure 8).

Navigate to Styles, and create a new style called `real_estate_coverage`. Add the code from Listing 1, validate and save (Figure 9). Navigate to Layers, `nyc_buildings` and click on the publish tab. Select `real_estate_coverage` and save (Figure 10). Previewing the layer will show Band1, 2 and 7 in different colours relating to area (Figure 11). Keep adding rules and bands until sufficient granularity is present, using the values in Table 2. This will result in the map shown in (Figure 12).

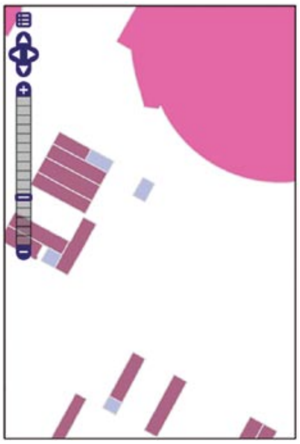
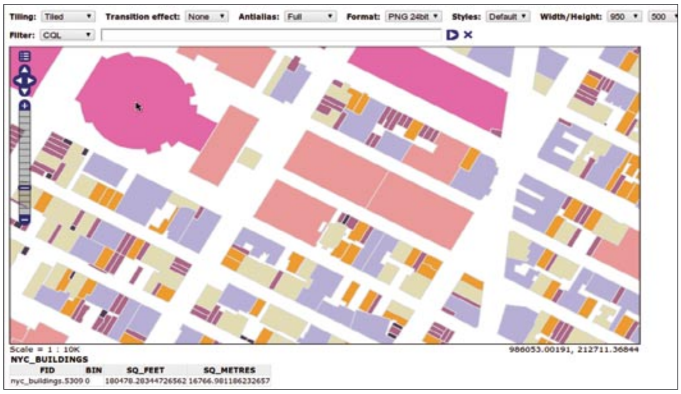


Figure 11. *Openlayers map with sample code*



Tiling: Tiled Transition effect: None Antialias: Full Format: PNG 24bit Styles: Default Width/Height: 950 500
Filter: CQL

Scale = 1 : 10K
986053.00191, 212711.36844

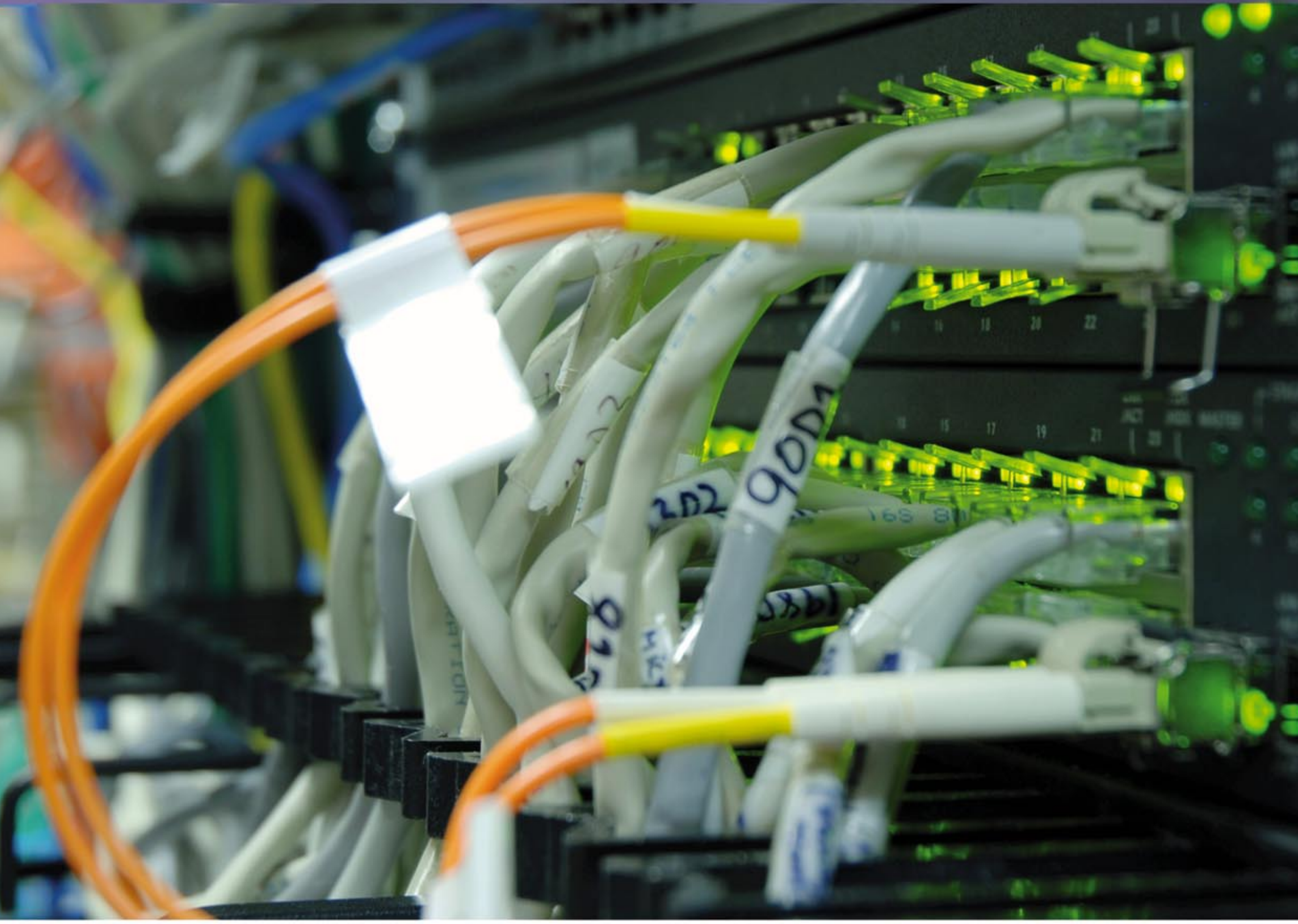
NYC_BUILDINGS

FID	BIN	SQ_FEET	SQ_METRES
nyc_buildings.5300	0	180478.20344726562	16766.981186232657

Figure 12. *The completed Openlayers map with SLD using all values from Table 2*

ROB SOMERVILLE

Rob Somerville has been passionately involved with technology both as an amateur and professional since childhood. A passionate convert to *BSD, he stubbornly refuses to shave of his beard under any circumstances. Fortunately, his wife understands him (she was working as a System/36 operator when they first met). The technological passions of their daughter and numerous pets are still to be revealed.



Collectd

A look at the Systems Statistics Collection Daemon

Systems Administrators need a variety of tools to properly monitor and tune their systems to the various loads.

What you will learn…

- Basic TCP/IP Networking
- How to use vi, emacs or other text editor
- Use CLI shell commands and compile programs

What you should know…

- Principle methodologies of Network Management Systems
- To build scalable statistics collection network
- How to capture data from different system resources

The typical open-source packages available today either monitor uptime on servers and applications, or monitor system metrics for performance. Nagios and Monit would be examples of the former. Cacti and Zabbix tend to lean toward the latter. These are just a few of many open source packages mentioned, and most can function in both areas.

Capturing performance metrics is typically done via SNMP (*Simple Network Management Protocol*) by a central *Network Management System* (NMS). The NMS server polls it's list of hosts (*clients*) collecting data from each client's *snmp* agent. Each client and server usually have to be configured for access and which node of the *snmp* tree the NMS server is allowed to read and/or optionally write. Each monitored client running an SNMP agent must process the resources (or subset of) and process any configured sub-agents before finally returning the result set of data to the NMS server.

SNMP in itself is an excellent protocol as it exposes either directly or indirectly, a vast set of statistical metrics on the given machine. Examples of these metrics wold be network interface values, individual cpu loads, memory, routing tables, system temperature, and disk resources.

Two major drawbacks exist in this type of *pull* methodology. First, the time and processing load, to *poll* each of the host's it's monitoring, increases as additional hosts and services are added. This can cause incrementally higher loads and longer polling cycles on the NMS server as additional clients are monitored. Usually these NMS systems poll at 1 or 5 minute polling intervals. This delay can mute or attenuate performance statistics and make network or system load spikes more difficult to troubleshoot.

Collectd

The Collectd package flips the conventional snmp *pull* methodology on it's proverbial ear, making the server *passive* allowing the clients to *push* statistic data up. This methodology

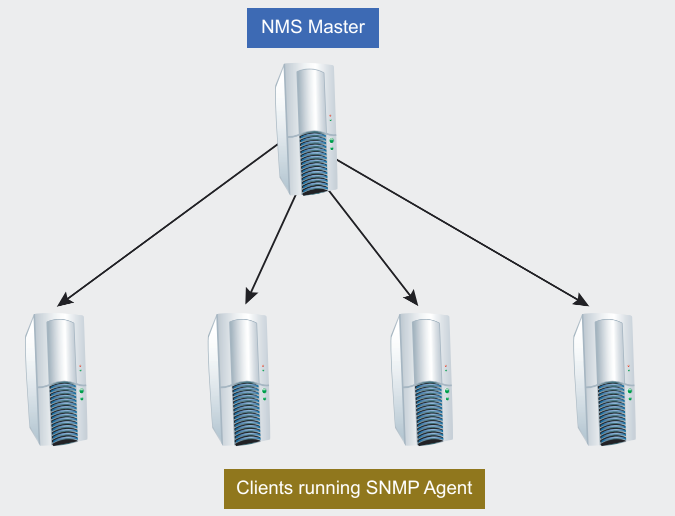


Figure 1. *NMS-snmp.png, Typical SNMP / NMS configuration*

scales well, while only minimal overhead is added at the client level. Collectd's ability to sample on very short intervals allows for high resolution statistics and graphs of a particular resource. Sampling intervals as short as 10 seconds can give a very detailed view of a servers performance, such as this CPU graph (see Figure 3). This is just one of Collectd's features that make it a powerful platform for collection of performance data. Here are a few favorites:

- Short sampling intervals (default is 10 seconds)
- Zero server side configuration possible
- Multicast or unicast network transport
- Local storage of collected data optional, in CSV, or RRD formats
- Server can be a client and relay to remote Collectd servers.
- Builds easily
- Variety of plug-ins ready made
- Scales easily to hundreds of clients
- Low overhead on client operating systems
- Scriptable data collection via plugins for Perl, and Python
- Small UDP or TCP network payload traffic from clients

The complete feature list as well as the long list of available plug-in's can be found at the Collectd website: *http://collectd.org/features.shtml*.

The website clearly states that Collectd is a framework or platform, and like it's name, it *collects* data. What it does not do is present data in a visual manner. That will be addressed later in the series.

Setting up a Collectd Server

To begin, setup the *Server* system which will collect and accumulate statistics sent to it by the unicast and multi-cast clients. The multi-cast IP configuration is recommended as it is simple to setup and allows the greatest flexibility. One can configure multiple servers on the same subnet to capture/process the statistics, without any changes on the client systems. Grab the latest daily snaphost tarball from *http://snapshots.tokkee.org/collectd/*. Copy the link from a browser then paste it into a ssh window..

```
eg: $ fetch http://snapshots.tokkee.org/collectd/current/
                     collectd-5.0.0.34.g514a9fe.tar.gz
```

Un-tar it in a place where it can be built.

```
   $ cd ~bharris
   $ tar zxvf collectd-5.0.0.34.g514a9fe.tar.gz
   $ cd collectd-5.0.0.34.g514a9fe/
```

Collectd defaults to `/opt`, which by default is on a typically small root filesystem, so.. build and install with `/usr/local` as the base:

```
   $ ./configure --prefix=/usr/local
   $ make
```

Optionally, it's possible to store the Collectd statistic data in RRD (*Round Robin Database*) files. Install *rrdtool* and build with the appropriate options listed below. RRD files are actual databases and have some additional capabilities that CSV (comma separated values) do not, such as creating some nice looking graphs, automated management and selection of dataset ranges.

```
   # pkg_add -r rrdtool
   # ./configure --prefix=/usr/local --enable-rrdtool
```

Now install it by su'ing to root

```
   $ su [ENTER]
   # make install
```

Edit `/usr/local/etc/collectd.conf` and enable a few things: Scroll down un-comment the csv plug-in:

```
##LoadPlugin cpufreq
LoadPlugin csv
##LoadPlugin curl
```

This enables the csv plugin. The default directory is defined farther down:

```
<Plugin csv>
        DataDir „/usr/local/var/lib/collectd/csv"
```

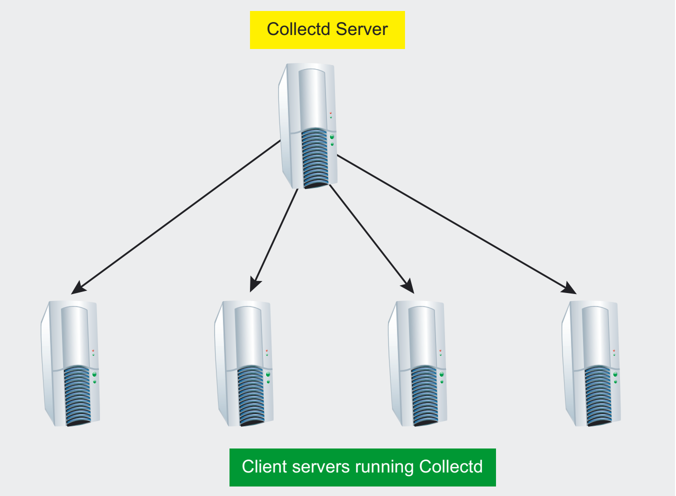


Figure 2. *Collectd-Master.png, Collectd client server topology*

```
#       StoreRates false
</Plugin>
```

To enable RRD database for output instead of or in addition to csv plug-in, enable it as well:

```
##LoadPlugin rrdcached
LoadPlugin rrdtool
##LoadPlugin sensors
...
<Plugin rrdtool>
        DataDir „/usr/local/var/lib/collectd/rrd”
        CacheTimeout 120
        CacheFlush   900
</Plugin>
```

Now configure the Collectd server's network plugin to listen on a multi-cast address:

```
#       # server setup:
#       Listen „ff18::efc0:4a42” „25826”   // Comment out
                    IPv6 if applicable
         Listen „224.0.0.1” „25826”    // local IPv4
                    multi-cast address
#       <Listen „239.192.74.66” „25826”>
```

Test the server configuration with:

```
   collectd -t  # tests the config file's syntax and exits
   collectd -T  # tests that plugins load, then exits.
```

If there is a problem, it will complain and suggest where to look. If all is well, start daemon...

```
   /usr/local/bin/collectd [ENTER]
```

Look in `/usr/local/var/lib/collectd/csv` and directories with the hostname of your local machine and sub-directories under it for the various resources, cpu, memory, interfaces, etc, should appear, as Collectd captures data on it the server itself.

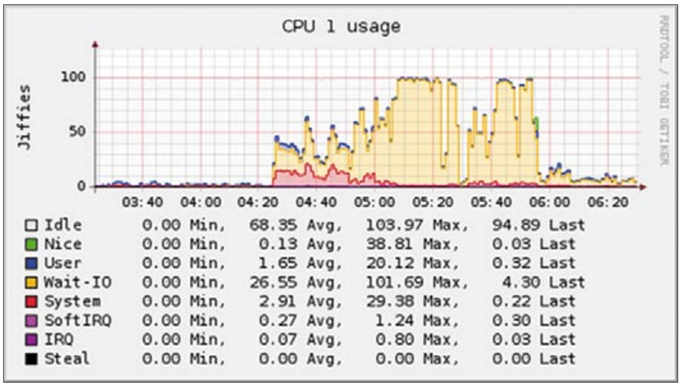


Figure 3. *Plugin-cpu.png, Hi Resolution CPU statistics*

Setting up a Collectd Client

Installing a client is exactly the same as a server, with even less configuration necessary.

On the first client system, download (or sftp) the Collectd tarball to the client machine and compile it using the same steps as for the server above.

Edit the Collectd configuration file (`/usr/local/etc/collectd.conf`). Modify the `<plugin network>` to use our multicast address and comment, out the IPv6 and unicast default addresses:

```
<Plugin network>
#       # client setup:
#       Server „ff18::efc0:4a42” „25826”          //
                  coment out the IPv6
        Server „224.0.0.1” „25826”         // put in our
                  multicast address
#       <Server „239.192.74.66” „25826”>          //
                  Coment out unicast
#               SecurityLevel Encrypt
#               Username „user”
#               Password „secret”
```

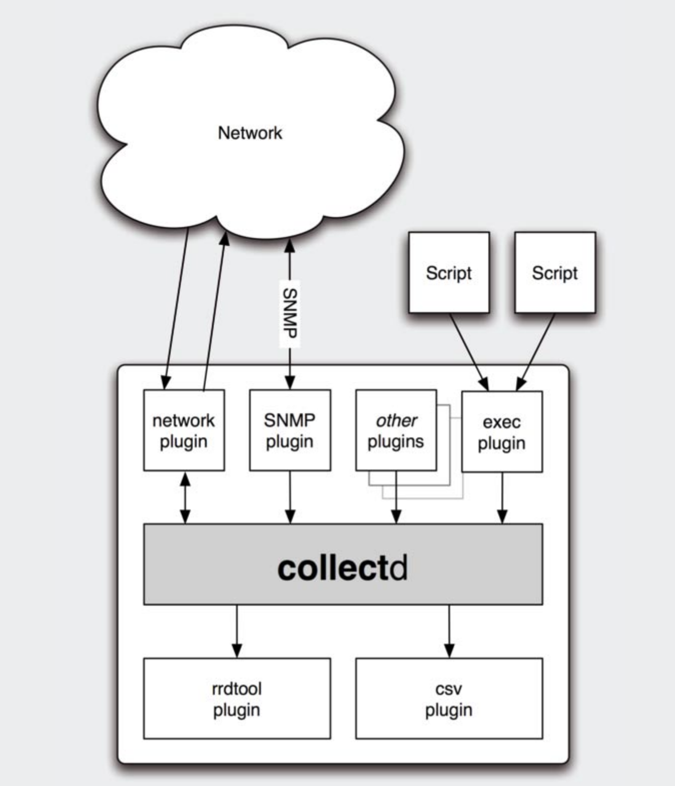


Figure 4. *Architecture-schematic.png, Collectd Architecture*

```
#               Interface „eth0”
#       </Server>                   // unicast
#       TimeToLive „128”
...
</Plugin network>
```

This tells the Collectd *client* to push it's statistics data out onto the local multicast 224.0.0.1 where our server instance of Collectd will pick up the packets. Start the Collectd daemon on the client machine, and a directory of the client's hostname should appear on the Server machine in the csv directory.

Adding additional clients requires no configuration on the server side. Collectd does support authenticated access, which is well documented in the configuration file and on the website. The server will soon build a directories containing comma delimited files for each resource of the client such as:

```
/usr/local/var/lib/collectd/csv/alix3]$ ls -l
total 14
drwxr-xr-x  2 root  wheel  512 Jun 25 14:54 cpu-0
drwxr-xr-x  2 root  wheel  512 Jun 25 14:54 interface-ath0
drwxr-xr-x  2 root  wheel  512 Jun 25 14:54 interface-lo0
drwxr-xr-x  2 root  wheel  512 Jun 25 14:54 interface-vr0
drwxr-xr-x  2 root  wheel  512 Jun 25 14:54 interface-wlan0
drwxr-xr-x  2 root  wheel  512 Jun 25 14:54 load
drwxr-xr-x  2 root  wheel  512 Jun 25 14:54 memory
```

Each directory will have a date stamped file for each resource in a format easily parsed.

```
$ ls -l load
total 40
-rw-r--r--  1 root  wheel  39975 Jun 25 17:33 load-2011-06-25
   epoch,shortterm,midterm,longterm
1309031682.894,0.000000,0.000000,0.000000
1309031690.461,0.000000,0.000000,0.000000
1309031700.463,0.000000,0.000000,0.000000
1309031710.461,0.000000,0.000000,0.000000
```

Unicast Versus Multicast

There are server deployments where machines are physically located close, such as on the same subnet, where multi-cast configurations would work well. When servers are remote distances from each other, or on routed subnets that prevent multi-cast packet traversal, unicast is the correct transport.

There are configurations where one master Collectd services a group of servers, as in a remote office, then it relays via unicast to a remote master back at headquarters, thereby using both transports efficiently.

Plug-ins

Collectd's configure script will enable all plug-ins that it is able to find necessary dependencies for. If a particular plug-in would meet specific need, such as snmp queries to a router or printer, then the supporting packages will need to be installed and Collectd built with those options enabled. The website and package documentation provide directions on how to accomplish this.

The Collectd Plug-in architecture is a powerful structure that enable Collectd to extend beyond typical SNMP and NMS boundaries. Cacti for example required several server scripts and device types to be defined, just to add a data element from an SNMP agent. One simply formats the data and feeds it into Collectd's EXEC plug-in, or scripted via Python or Perl plug-in.

Sending application metrics, the local temperature, or any number of other useful values, is straightforward in the Collectd's plug-in system. A simple example would be to use the builtin DNS plug-in to collect name traffic statistics..

```
   # ./configure --prefix=/usr/local --enable-dns
```

and then enable it in the Collectd configuration file `/usr/local/etc/collectd.conf`

```
##LoadPlugin disk
LoadPlugin dns
#LoadPlugin email

<Plugin dns>
      Interface „vr0”            // your lan interface
#       IgnoreSource „192.168.0.1”
#       SelectNumericQueryTypes true
</Plugin>
```

Good Steward

As an agent, Collectd consumes very little in the way of cpu, since the majority of the statistics it gathers, it does so directly by reading kernel values, or indirectly via as library calls. This allows the agent to send it's timely statistics, yet consume little in resources. The server, likewise has very low overhead.

Now that Collectd is capturing statistics, the next article in the series will demonstrate some different methods of presenting the statistics visually.

BILL HARRIS

Bill Harris is a Unix Sys Admin in the North Texas area. He enjoys building FreeBSD and Ubuntu servers at home. savoy9020@gmail.com

Using Memcached

for High Scalability Web Services

Been creating web services/ applications for 6 years and until recently decided to try out memory caching technology instead of hitting up the ole SQL server for the same records over and over again.

What you will learn…

- installation of Memcached on FreeBSD
- setup Memcached for PHP
- how to use Memcached

What you should know…

- basic knowledge of FreeBSD ports
- how to edit files
- basic understanding of PHP

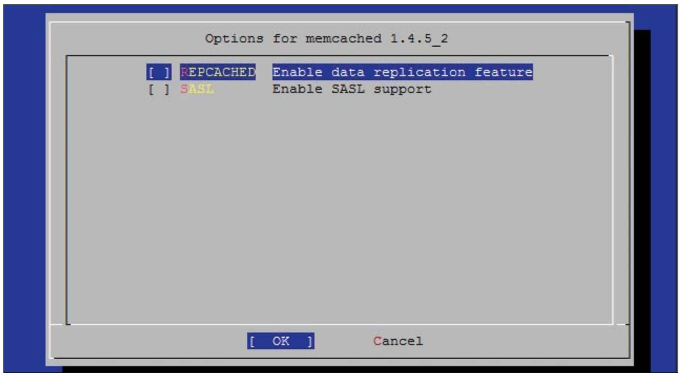
Memcached is a *high-performance, distributed memory object caching system* which allows data to be saved in-memory as key-values.

Installing and running Memcached on FreeBSD is pretty straight-forward. You can have a scalable web service running on Memcached in no time at all. *In this article you will need a server with Apache 2.x and PHP 5.3.x installed.*

Installing Memcached via FreeBSD Port

```
# cd /usr/ports/databases/memcached
# make install clean (Figure 1)
```

Wait for installation to complete.



```
Options for memcached 1.4.5_2

[ ] REPCACHED   Enable data replication feature
[ ] SASL        Enable SASL support
```

Figure 1. *Install Memcached (Customize Options) according to your needs*

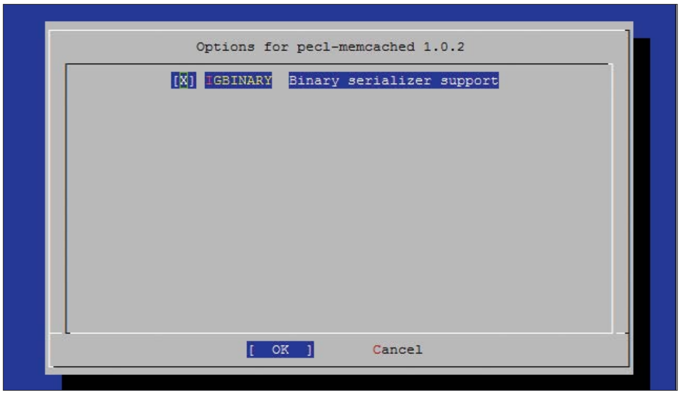
Installing PHP Memcached Extension

Next we will install the much needed PHP Extension for Memcached *pecl-memcached:*

```
# cd /usr/ports/databases/pecl-memcached
# make install clean (Figure 2)
Wait for installation to complete. Restart web server.
# /usr/local/etc/rc.d/apache22 restart
```

Verify Installation of PECL-Memcached

```
# ee /usr/local/etc/php/extension.ini
Make sure that "extension=memcached.so" is listed as an
installed extension. (Figure 3)
```



```
Options for pecl-memcached 1.0.2

[X] IGBINARY  Binary serializer support
```

Figure 2. *PECL-Memcached extension is necessary for PHP integration*

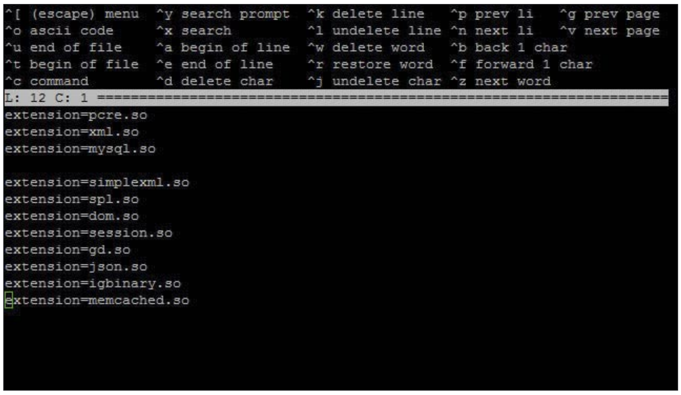
Listing 1. *Simple PHP Script*

```
<?php
############################
# SIMPLE MEMCACHED SAMPLE
# DIEGO MONTALVO 062311
############################
$mc = new Memcached();
$mc->addServer("localhost", 11211);//NEEDED FOR
                    MEMCACHED CONNECTION

$my_key = 'first_key';
$my_value = 'my first saved memcached key!';

//SET KEY-VALUE INTO MEMCACHED
$set = $mc->set($my_key, $my_value, time() + 120);//
                    EXPIRES EVERY 2 MINUTES

echo $mc->get('first_key');
?>
```

In order for Memcached to start on bootup you must edit the `/etc/rc.conf` file and enable it.

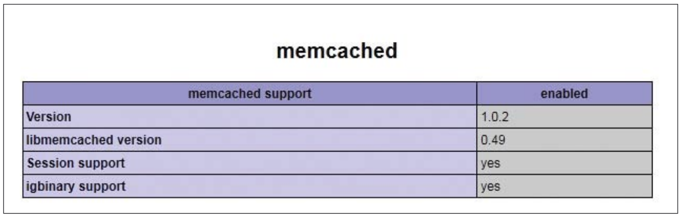
```
# ee /etc/rc.conf
```

Add the following line:



```
^[ (escape) menu  ^y search prompt  ^k delete line   ^p prev li   ^g prev page
^o ascii code     ^x search         ^l undelete line ^n next li   ^v next page
^u end of file    ^a begin of line  ^w delete word   ^b back 1 char
^t begin of file  ^e end of line    ^r restore word  ^f forward 1 char
^c command        ^d delete char    ^j undelete char ^z next word
L: 12 C: 1 ====================================================================
extension=pcre.so
extension=xml.so
extension=mysql.so

extension=simplexml.so
extension=spl.so
extension=dom.so
extension=session.so
extension=gd.so
extension=json.so
extension=igbinary.so
extension=memcached.so
```

Figure 3. *Make sure memcached.so is listed in extension.ini*



memcached

memcached support	enabled
Version	1.0.2
libmemcached version	0.49
Session support	yes
igbinary support	yes

Figure 4. *Verify Memcached is installed as a PHP extension on your web browser*

On the 'Net

- *http://memcached.org/* – Official site
- *http://php.net/manual/en/book.memcache.php* – PHP Memcached Extension

```
Memcached_enabled = "YES"
Save and exit.
```

Next we will verify proper installation of `pecl-memcached`. Inside your web server directory create a file `test.php` and add the following: `<?php phpinfo(); ?> Save and exit.`

In your web browser point URL to where `test.php` is stored, `memcached` should be listed as an installed extension. (Figure 4)

Testing Memcached Installation

To check if the Memcached extension is loaded in PHP, execute the following command.

```
# php -i | grep memcached
Successful Output ›
memcached
memcached support => enabled
libmemcached version => 0.49
Registered save handlers => files user memcached
PWD => /usr/ports/databases/pecl-memcached
_SERVER[„PWD"] => /usr/ports/databases/pecl-memcached
_ENV[„PWD"] => /usr/ports/databases/pecl-memcached
```

Using Memcached will greatly speed up your database or API driven web service. Memcached like anything else will take some tweaking according to your wants and needs. For the most part you now have a starting point with this article on how to install, setup and run Memcached on your server. Welcome to the world of highly-scalable web services using Memcached. Finally you and your dynamic driven service can chill out.

DIEGO MONTALVO

Diego Montalvo is a web/ mobile application developer which has developed many web services / mobile applications such as buildasearch.com, bobchatter.com, urloid.com and many others. Diego resides in San Diego, California, but is currently tasting the best Tequilas in Guadalajara, Mexico. Feel free to contact Diego at diego@earthoid.com

LDAP Authentication and Authorization

Of Unix Users Under OpenBSD

Unlike most Unix-like operating systems, OpenBSD does not come with PAM nor nsswitch which made it tedious to authenticate local users against a remote database like LDAP. That was until ypldap(8) came along.

What you will learn…

- Manage OpenBSD system users in LDAP
- Some concepts and daemons specifics to OpenBSD (bsd_auth(3), ldapd(8), ypldap(8))

What you should know…

- Basic Unix administration knowledge
- LDAP concepts
- PAM / NSS authentication
- YP / NIS

This article will focus on the plumbing much more than on the theory as we want to get things done and not go into details about all the technologies involved.

Since LDAP could be the topic of an entire book, it is assumed that your are comfortable with its concepts and usage; former OpenLDAP experience is strongly encouraged.

The BSD Authentication system

OpenBSD made the choice of using `bsd_auth(3)` over PAM(3) (Linux-PAM on Linux, OpenPAM on FreeBSD and NetBSD). `bsd_auth(3)` was inherited from BSDi (aka BSD/OS, a BSD/386 proprietary version) and differs from PAM(3) on several levels.

`bsd_auth(3)` does authentication using setuid `login_*` helpers under `/usr/libexec/auth/` which can only be run by a separate process that is part of the auth group. This prevents brute-force attacks and implements privilege separation.

Several helpers are available on a default OpenBSD installation (password file, Kerberos, S/Key...). We will see how to use LDAP as an authentication source using the regular `login_passwd(8)` helper and `YP(8)` maps retrieved from `ypldap(8)`.

As mentioned, while `bsd_auth(3)` provides similar functionality as PAM for authentication, authorization can only come from either `/etc/master.password(5)` or `YP(8)`. The reason is that there is no Name Service Switch support on OpenBSD due to the fact that it would require loading dynamic modules in libc to get user information which was not considered acceptable by the project.

LDAPD

OpenBSD comes with its own LDAP server implementation known as `ldapd(8)` (written by Martin Hedenfalk aka *martin@*). It is not meant to compete directly with OpenLDAP but to provide a lightweight, simple and secure alternative for most needs. Since it comes with the base system, we won't need to install any third-party applications to make use of it (of course OpenLDAP could be used instead).

Configuration

`ldapd(8)` provides the following schemas under `/etc/ldap`: `core.schema`, `inetorgperson.schema` and `nis.schema` which are enough for setting up user authentication.

Configuration is done in `/etc/ldapd.conf(5)`. First we are going to setup our *base DN* (*Distinguished Name* – usually derived from the domain name) and a privileged user.

As always with OpenBSD, the configuration already comes with sensible defaults, all we need to do is uncomment some lines and edit them accordingly.

For the sake of the example we'll use the *bsdmag.org* domain. Most entries should look familiar to OpenLDAP users.

Let's have a look at rootpw; here we want to associate the LDAP Manager with the root password and use `bsd_auth(3)` which allows us to authenticate via simple bind, SASL-PLAIN. There is no point in creating a specific *Manager* entry in the database itself and it becomes handy when the root password changes since nothing needs to be done on the LDAP side.

Starting the server

Let's now start `ldapd(8)` in foreground and verbose mode to check our configuration is correct. So far so good... Let's just `<ctrl-c>` and run `ldapd(8)` in normal mode:

```
# ldapd
```

Add an entry to `rc.conf.local(8)` so that it is started automatically on boot:

```
# echo "ldapd_flags=" >> /etc/rc.conf.local
```

Populating the LDAP database

Previously I stated that we wouldn't need to install any external applications since we were going to use `ldapd(8)` and `ypldap(8)` which are both included in the base system. While this is true for the server side, OpenBSD still misses an LDAP client utility, so we will need to install the one from OpenLDAP to be able to interact with our server.

To do so, edit your `PKG_PATH` or `pkg.conf(5)` accordingly then:

```
$ sudo pkg_add openldap-client
```

Listing 1. */etc/ldapd.conf*

```
schema "/etc/ldap/core.schema"
schema "/etc/ldap/inetorgperson.schema"
schema "/etc/ldap/nis.schema"

listen on lo0
listen on "/var/run/ldapi"

namespace "dc=bsdmag,dc=org" {
        rootdn          "cn=Manager,dc=bsdmag,dc=org"
        rootpw          {BSDAUTH}root
        index           sn
        index           givenName
        index           cn
        index           mail
}
```

Now we have all the pieces to populate the database. To do so, we need to create an LDIF (*LDAP Data Interchange Format*) file containing the root of our namespace and the branch were we would be adding users (i.e. *People*). Fire up your favorite editor and create a base.ldif file like the one below. Note that all the entries are declared in the corresponding schemas under `/etc/ldap/` and describing them here would be out of scope.

Let's add the entry. By default, `ldapd(8)` requires a secure connection so we will communicate over the Unix socket (`/var/run/ldapi`) which is considered as such. It seemed to have worked (see Listing 3 and 4).

The procedure is the same to add a new user so let's add one. We will call him *John Joe* (jdoe) with a *UID* of 2000 and a *GID* of 10 (which corresponds to the users group in OpenBSD).

All entries are pretty standard except for userPassword which value is not typical for LDAP. Usually LDAP passwords look like `{SSHA}XXX or {CRYPT}XXX` etc. Here we give it a regular *Blowfish* encrypted password, as it is the default for OpenBSD `master.passwd(5)`. In effect the `YP(8)` passwd map will be exactly the same as one provided by `ypserv(8)`.

To create a Blowfish password hash, do the following (replace *secret* with the desired password):

Listing 2. *Running ldapd(8) in debug and verbose mode*

```
# ldapd -dv
Jun 20 08:39:03.292 [11952] parsing config /etc/
                    ldapd.conf
Jun 20 08:39:03.293 [11952] parsing schema file '/etc/
                    ldap/core.schema'
Jun 20 08:39:03.295 [11952] parsing schema file '/etc/
                    ldap/inetorgperson.schema'
Jun 20 08:39:03.296 [11952] parsing schema file '/etc/
                    ldap/nis.schema'
Jun 20 08:39:03.297 [11952] parsing namespace
                    dc=bsdmag,dc=org
Jun 20 08:39:03.297 [11952] startup
Jun 20 08:39:03.299 [6680] listening on /var/run/ldapi
Jun 20 08:39:03.300 [6680] listening on fe80:4::1:389
Jun 20 08:39:03.300 [6680] listening on ::1:389
Jun 20 08:39:03.300 [6680] listening on 127.0.0.1:389
Jun 20 08:39:03.300 [6680] opening namespace
                    dc=bsdmag,dc=org
Jun 20 08:39:03.300 [6680] ldape: entering event loop
```

```
$ encrypt -b 8 - secret
$2a$08$xXtwFN2ClWPTdv3nlE0SW.10VVz4Bz5NGoygNtmkVIaBMOLtmehmK
```

Let's add the user... (see Listing 5 and 6).

I encourage you to read the ldapd(8), ldapd.conf(5) and ldapctl(8) manual pages as they contain precious information for tuning (indexes), securing (ACLs) and interacting with your LDAP server.

Our database is now fully populated with all the required information, so let's move to the next step...

YPLDAP

As the manual page states, ypldap(8) is a daemon providing YP(8) maps using LDAP as a back-end The YP(8) approach was chosen because it wouldn't require any change in libc or the way user information is gathered. That means ypldap(8) will act as a replacement for ypserv(8).

Configuration

ypldap(8) doesn't come with a default configuration file, but the man page contains an example that we can copy/paste to /etc/ypldap.conf(5) then adapt to our needs.

Note that we will only provide *passwd* and not *group* maps because ypldap(8) lacks support for netid(5) and consequently getgrouplist(3). More information in the *Drawbacks and limitations* section.

So let's create ypldap.conf(5) as follow (see Listing 7). You can refer to the corresponding manual page to know about all the possible options (since it contains our Manager password, *make sure the file mode is 0600* to that only root can read and write to it).

The mapping of attributes should be fairly easy to understand and is used to map an LDAP entry to a master.passwd(5) field. For example, in our setup we will use the uidNumber from the LDAP user as its *uid*. Some attributes can also be hard-coded whatever value is set in LDAP; e.g. we want the account expiration and password change times to never expire and we set the login(1) class to empty (which is the *default* class). Let's check our configuration.

```
# ypldap -n
configuration OK
```

Good, so let's start ypldap(8) in debug and verbose mode to have a better idea of what it does. Since the YP(8) subsystem is built on RPC (*Remote Procedure Call*), we also need to start RPC bind daemon: portmap(8).

Listing 3. *base.ldif*

```
dn: dc=bsdmag,dc=org
objectClass: top
objectClass: dcObject
objectClass: organization
dc: bsdmag
o: Root of the organization

dn: ou=People,dc=bsdmag,dc=org
objectClass: top
objectClass: organizationalUnit
ou: People
description: LDAP users
```

Listing 4. *Creating the LDAP tree base*

```
$ ldapadd -D cn=Manager,dc=bsdmag,dc=org -H ldapi://
                    %2fvar%2frun%2fldapi -W < base.ldif

adding new entry "dc=bsdmag,dc=org"

adding new entry "ou=People,dc=bsdmag,dc=org"
```

Listing 5. *jdoe.ldif*

```
dn: uid=jdoe,ou=People,dc=bsdmag,dc=org
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
objectClass: account
cn: John Doe
uid: jdoe
uidNumber: 2000
gidNumber: 10
homeDirectory: /home/jdoe
userPassword: $2a$08$xXtwFN2ClWPTdv3nlE0SW.10VVz4Bz5NG
                    oygNtmkVIaBMOLtmehmK
loginShell: /bin/ksh
```

Listing 6. *Adding a user to LDAP*

```
$ ldapadd -xD cn=Manager,dc=bsdmag,dc=org -H ldapi://
                    %2fvar%2frun%2fldapi -W < jdoe.ldif
Enter LDAP Password: <input your root password>

adding new entry "uid=jdoe,ou=People,dc=bsdmag,dc=org"
```

It looks like our password line is pushed correctly (see Listig 8) :-) So now we can just run in normal mode. Just `<ctrl-c>` the currently foregrounded `ypldap(8)` process then:

```
# ypldap
```

Let's add an entry to `rc.conf.local(8)` so that `portmap(8)` is started automatically on boot:

```
# echo “portmap=” >> /etc/rc.conf.local
```

Add the Listing 9 bits to `rc.local(8)` so that it is automatically started as well (there is no `rc.conf(8)` support for `ypldap(8)` yet).

Setting up a YP(8) client

Now that we have all the required server-side services running, let's see how to configure client authentication. The following procedure needs to be repeated for each machine willing to use LDAP as an authentication source.

Since `ypldap(8)` acts as a `YP(8)` server daemon we just need to setup our machine as a regular `YP(8)` client.

Listing 7. */etc/ypldap.conf*

```
domain    "bsdmag.org"
provide map      "passwd.byname"
provide map      "passwd.byuid"

directory "127.0.0.1" {
   binddn "cn=Manager,dc=bsdmag,dc=org"
   bindcred "root_password"
   basedn "ou=People,dc=bsdmag,dc=org"

   passwd filter "(objectClass=posixAccount)"

   attribute name maps to "uid"
   attribute passwd maps to "userPassword"
   attribute uid maps to "uidNumber"
   attribute gid maps to "gidNumber"
   attribute gecos maps to "cn"
   attribute home maps to "homeDirectory"
   attribute shell maps to "loginShell"
   fixed attribute change "0"
   fixed attribute expire "0"
   fixed attribute class ""

   group filter "(objectClass=posixGroup)"
```

As seen above, we will only configure `YP(8)` for users management, not groups.

First, let's set our default `YP(8)` domain and at it to the `defaultdomain(5)` file so that it gets set automatically on boot.

```
# domainname bsdmag.org
# echo “bsdmag.org” > /etc/defaultdomain
```

While `ypbind(8)` can broadcast to find the server providing the *bsdmag.org* maps it's considered a best practice to specify the servers under `/etc/yp/domainname`. Here we'll establish a binding with localhost

```
# mkdir /etc/yp
# echo “localhost” > /etc/yp/bsdmag.org
```

The last step is to instruct the system to use the `YP(8)` maps as an additional source for authentication. To do so, we need to add the following line at the end of `master.passwd(5)` using the `vipw(8)` utility:

```
   +:*::::::::
```

We can now start `ypbind(8)`, the client side of the `YP(8)` subsystem. Note that when defaultdomain(5) exists and is configured, `ypbind(8)` will start automatically on boot without the need for an entry in `rc.conf.local(8)`.

```
# ypbind
```

Listing 8. *Starting ypldap(8) in foreground and verbose mode*

```
# portmap
# ypldap -dv
startup [debug mode]
configuration starting
applying configuration
connecting to directories
starting directory update
searching password entries
searching group entries
updates are over, cleaning up trees now
flattening trees
pushing line: jdoe:$2a$08$xXtwFN2ClWPTdv3nlE0SW.10VVz4
                    Bz5NGoygNtmkVIaBMOLtmehmK:2000:10:
                    :0:0:John Doe:/home/jdoe:/bin/ksh
```

Now comes the moment of truth... let's check if our system can see the LDAP users. According to Listings 10 and 11, this is a success! All that is left to do now is to provide a home directory for that user. Since we are not using `PAM(8)`, we cannot use modules like `pam_mkhomedir` to automatically create non existing home directories when a users log in, so let's do it manually.

```
# cp -Rp /etc/skel /home/jdoe
# chown -R jdoe:users /home/jdoe
```

We should now be able to login.

Secure communications

To accept simple binds (plain text passwords), ldapd(8) must consider the connection as secure, otherwise it will refuse to answer (*refusing non-anonymous bind on insecure connection*). There are three ways to this:

- communicate over the `/var/run/ldapi` socket
- add the secure keyword at the end of the *listen on if* entry in `ldapd.conf(5)`
- use `SSL(8)` encryption

In our sample setup since both `ypldap(8)` and `ldapd(8)` were running on the same system, we used the `/var/run/ldapi` socket to connect to the LDAP service.

But if we want to split the services between several machines, we need to make them communicate in a secure way and we'll use `SSL(8)` for that. Note that encryption will only happen between `ldapd(8)` and `ypldap(8)`: communication between `YP(8)` clients and `ypldap(8)` will still be *unencrypted*.

Listing 9. */etc/rc.local*

```
echo -n ' ypldap'; /usr/sbin/ypldap
```

Listing 10. *LDAP user seen as a local one*

```
# user info jdoe
login   jdoe
passwd  $2a$08$xXtwFN2ClWPTdv3nlE0SW.10VVz4Bz5NGoygNtm
                  kVIaBMOLtmehmK
uid     2000
groups  users
change  NEVER
class
gecos   John Doe
dir     /home/jdoe
shell   /bin/ksh
expire  NEVER
```

`ldapd(8)` can use *tls* (on regular LDAP port 389/TCP) or *ldaps* (636/TCP) along with SSL(8) certificates. The creation of such certificates is covered by so many documents that there would be no point in repeating the procedure here; see `starttls(8)` for more information and examples.

Drawbacks and limitations

Now that we have everything running, let's investigate some common problems and how to resolve or mitigate them when possible.

Listing 11. *Login to a local OpenBSD console*

```
login: jdoe
Password:
OpenBSD 4.9-current (GENERIC.MP) #81: Mon Jun 20 13:
                   56:42 MDT 2011

Welcome to OpenBSD: The proactively secure Unix-like
                   operating system.

Please use the sendbug(1) utility to report bugs in
                   the system.
Before reporting a bug, please try to reproduce it with
                   the latest
version of the code.  With bug reports, please try to
                   ensure that
enough information to reproduce the problem is
                   enclosed, and if a
known fix for it exists, include that as well.

$ ls -al

drwxr-xr-x  3 jdoe  users  512 Jun 20 21:35 .
drwxr-xr-x  6 root  wheel  512 Jun 22 12:08 ..
-rw-r--r--  1 jdoe  users   22 Apr 28  2009 .Xdefaults
-rw-r--r--  1 jdoe  users  773 Feb  2  2009 .cshrc
-rw-r--r--  1 jdoe  users  398 May 14  2009 .login
-rw-r--r--  1 jdoe  users  113 Apr 25  2009 .mailrc
-rw-r--r--  1 jdoe  users  218 Nov 29  2006 .profile
drwx------  2 jdoe  users  512 Dec  6  2009 .ssh
```

Passwords

The approach we have taken in our sample setup has an obvious drawback: as in a regular `YP(8)` setup, the user password *hash* is transferred in clear-text over the wire (and can be seen by running *getent passwd*). Besides we will not be able to use this password to bind to the LDAP server directly (this can be considered a feature) which means the user will only be able to login on an OpenBSD box (PAM(8) requires an LDAP bind).

If you want a more standard and portable approach and to prevent the hash to be visible, you'll want to install the `login_ldap` package, set it up accordingly (enable tls/ssl) and use the `slappasswd(8)` utility from OpenLDAP to create a standard LDAP password hash. The `ypldap(8)` configuration would also need to be modified as follow:

```
fixed attribute passwd „*” instead of attribute passwd maps
                       to „userPassword”
fixed attribute class „ldap” instead of fixed attribute class „”
```

A new `login.conf(5)` class named *ldap* will need to be created with an *auth-defaults* set to *login_-ldap*.

The reason we did not do this is because login_ldap isn't part of the base system (it depends on the OpenLDAP libraries) and we wanted to keep things as simple as possible.

Groups

As we've seen, `ypldap(8)` does not support secondary groups yet. This is one of the major limitations for now and there is no easy way to workaround this issue.

Until `netid(5)` support is implemented, `/etc/group(5)` needs to be managed locally.

Fail-over

`ldapd(8)` has no support for replication (multi-master mode or master – slave). Again there is no easy way around this, except using OpenLDAP.

If no `ypldap(8)` server is reachable, `ypbind(8)` will hang and no authentication / authorization will be possible anymore for anyone, even local users. That is why running several `ypldap(8)` servers is advised.

There is a safeguard by using the `/etc/netid(5)` file to allow certain users to log in even when YP(8) isn't available. To make sure the root user can login at any time, add the line from Listing 12.

Listing 12. */etc/netid*

```
unix.root@bsdmag.org 0:0,2,3,4,5,20,31
```

You can also add other users but keep in mind that each user listed in this file will also need to exist locally in the `master.passwd(5)` file.

Boot order

`ypldap(8)` needs to run *after* the LDAP service is up and running. In a regular `ldapd(8)` setup, `rc(8)` will do the right thing and start daemons in the correct order.

However, when using OpenLDAP on the server side instead of `ldapd(8)`, you need to make sure it is started before `ypldap(8)`. That means you will need to manually add the `slapd(8)` startup command under `/etc/rc.local(8)` before the `ypldap(8)` one.

Conclusion

As we've seen, setting up LDAP authentication is pretty straightforward but that supposes that the whole infrastructure is running OpenBSD. The vast majority of LDAP setups for Unix authentication are based on Linux and expect PAM and NSS for direct server binds. OpenBSD clients do not work this way, they need a YP(8) server. To workaround this and to allow the integration of an OpenBSD workstation or server in the environment, all that is needed is to setup a `ypldap(8)` service on the machine itself, configure it to bind to the existing LDAP server and setup `ypbind(8)` to use localhost. Obviously, if several OpenBSD installations are planned on a mixed setup, the best is to dedicate one (or more) to run `ypldap(8)`.

While using `YP(8)` does have its shortcomings, it also brings a lot of simplicity in setting up users to authenticate against LDAP since all that is required is to turn the workstation into a `YP(8)` client.

Note that to keep the examples shown in this article as simple as possible all components were put on the same machine. Obviously, the `ypldap(8)` and `ldapd(8)` services can be split between different servers and listen on more interfaces than just localhost and/or a Unix socket. As mentioned several `ypldap(8)` daemons should run concurrently in a LAN to provide redundancy.

ANTOINE JACOUTOT

Antoine Jacoutot is an OpenBSD committer who lives in Paris, France. He is responsible for more than 300 packages, wrote the sysmerge(8) utility and is part the OpenBSD rc.d(8) framework development. He runs OpenBSD for pretty much everything.

Building a Complete Intrusion Detection System

with Snorby on BSD

FreeBSD and OpenBSD are a popular choice for installing the renovned open-source Snort intrusion detection. Documents have been written in the past for popular analysis tools such as BASE and Sguil, however nothing extensive has been created for Snorby.

What you will learn…

- A systematic approach to auto-configure the installation of Snorby.

What you should know…

- Basic FreeBSD knowledge to navigate the command line and the ports tree
- Basic knowledge of network security.

Snorby is a Ruby on Rails application built for network security monitoring interfacing Snort to provide network intrusion analysts with a variety of tools and graphs to investigate incidents. Snorby provides an impressive interface with the functionality required for monitoring a network for intrusion attempts. This article describes a complete step-by-step installation of Snorby using the necessary tools on FreeBSD.

What is not covered...

This how-to provides step-by-step instructions for how to get Snorby and Snort up and running on FreeBSD. The passwords used are default passwords included with Snorby to make things easier for new users. This configuration is not secure and should not be run in a production environment. Security administrators are always looking for information that is presented in a meaningful way. In the case of intrusion detection systems, this format can come in the form of a single flat file, a packet capture, or a database system with a certain amount of storage for analysis. There are several intrusion detection systems that are BSD friendly (Snort, Suricata, and BroIDS), however for this article the procedures will focus on using Snort in IDS mode.

Snort

http://www.snort.org. Snort is a freely available open-source intrusion detection and prevention system (IDS/IPS). For the purpose of this how-to, Snort will be configured with the default settings. This includes running Snort using only one capturing network interface.

Snorby

http://www.snorby.org. The next step is to install and configure Snorby. Snorby is an elegant and easy to use interface developed by Dustin Webber for investigating intrusion events. The interface is a Ruby on Rails application which can run as a standalone daemon, or as a module within the Apache web server. For this how-to, Snorby will be added to Apache with some additional configurations.

The first thing that needs to be completed is an install of FreeBSD. The machine or virtual machine used should have at least 512MB of memory to make sure Snorby runs well. All of the steps listed were performed on a Virtual Machine with FreeBSD i386 minimal install with the ports tree (See FREEBSD-INSTALL for installation instructions). The steps in this how-to also assume a single network interface has been setup with a static IP address. For example, a similar line should be found in `/etc/rc.conf` for a static interface:

```
ifconfig_em0=”inet 192.168.1.155 netmask 255.255.255.0”
```

After completing the install and restarting the FreeBSD server, login as root or an under-privileged user then run `su -` to install the prerequisites (see Listing 1). Once the prerequisites are installed, Snort and DAQ can be downloaded from the *snort.org* website (see Listing 2). DAQ stands for Data Acquisition, which is an abstraction library that allows more flexibility with packet access.

The default settings for compiling Snort will work by running `./configure`, but in order to use the latest snort.conf, there are several important options that need to be compiled in including IPv6 and zlib. Listing 3 shows the configure options to be used when compiling Snort.

Snort is now installed in `/usr/local/bin/snort`. Several important Snort config files need to be copied over to `/usr/local/etc/snort` as well as the creation of several directories for Snort and barnyard2 as seen in Listing 4.

The final part of the Snort configuration is the creation of the configuration file. Snort loads this configuration at startup which defines different options to detect multiple protocol anomalies as well as the signatures Snort will use for detection. Listing 5 can be copied and pasted into a shell prompt (or the automated install script referenced later can be used) to generate the `snort.conf` file.

Listing 1. *The following steps use -DBATCH and assume the default settings for each port.*

```
cd /usr/ports/ftp/wget
make -DBATCH install clean
cd /usr/ports/textproc/flex
make -DBATCH install clean
cd /usr/ports/devel/pcre
make -DBATCH install clean
cd /usr/ports/net/libdnet/
make -DBATCH install clean
cd /usr/ports/www/apache22
make -DBATCH install clean
cd /usr/ports/devel/ruby-gems/
make -DBATCH install clean
cd /usr/ports/converters/ruby-iconv/
make -DBATCH install clean
cd /usr/ports/textproc/libxml2
make -DBATCH install clean
cd /usr/ports/textproc/libxslt
make -DBATCH install clean
cd /usr/ports/graphics/ImageMagick
make -DBATCH install clean
cd /usr/ports/databases/mysql55-server/
make -DBATCH install clean
chown mysql:mysql -R /var/db/mysql
cd /usr/ports/devel/lwp
make -DBATCH install clean
cd /usr/ports/www/p5-LWP-UserAgent-WithCache/
make -DBATCH install clean
cd /usr/ports/security/p5-Crypt-SSLeay
make -DBATCH install clean
```

Listing 2. *Downloading Snort 2.9.0.5 and DAQ 0.5 and compiling DAQ with the default settings.*

```
mkdir /usr/src/snort
cd /usr/src/snort
/usr/local/bin/wget http://www.snort.org/downloads/867
mv 867 snort.tar.gz
/usr/local/bin/wget http://www.snort.org/downloads/860
mv 860 daq.tar.gz
tar -xvf snort.tar.gz
tar -xzf daq.tar.gz
cd daq-0.5
./configure
make
make install
```

Listing 3. *Configuration and compiling Snort*

```
cd /usr/src/snort/snort-2.9.0.5
./configure --enable-ipv6 --enable-gre --enable-mpls
                    --enable-targetbased --enable-
                    decoder-prepocessor-rules \

--enable-react --enable-flexresp3

make
make install
```

Listing 4. *Additional configs for Snort and barnyard2*

```
cp /usr/src/snort/snort-2.9.0.5/etc/*.config /usr/
                    local/etc/snort/
cp /usr/src/snort/snort-2.9.0.5/etc/*.map /usr/local/
                    etc/snort/
cp /usr/src/snort/snort-2.9.0.5/etc/threshold.conf
                    /usr/local/etc/snort/
mkdir /usr/local/lib/snort_dynamicrules/
mkdir /usr/local/etc/snort
mkdir /usr/local/etc/snort/rules
mkdir /usr/local/etc/snort/so_rules
mkdir /usr/local/etc/snort/preproc_rules
mkdir /var/log/snort
mkdir /var/log/barnyard2
touch /usr/local/etc/snort/rules/local.rules
touch /var/log/snort/barnyard2.waldo
```

Listing 5a. *Creating the /usr/local/etc/snort/snort.conf configuration file (Note: the IP range for HOME_NET is set to a standard class C subnet. Change the HOME_NET range to your specific network range). This configuration is adapted from the official snort.conf from the registered ruleset available from Sourcefire at http://www.snort.org*

```
ipvar HOME_NET [192.168.1.0/24]
ipvar EXTERNAL_NET [!$HOME_NET]
ipvar DNS_SERVERS $HOME_NET
ipvar SMTP_SERVERS $HOME_NET
ipvar HTTP_SERVERS $HOME_NET
ipvar SQL_SERVERS $HOME_NET
ipvar TELNET_SERVERS $HOME_NET
ipvar SSH_SERVERS $HOME_NET
portvar HTTP_PORTS [80,311,591,593,901,1220,1414,1830,2301,2381,2809,3128,3702,5250,7001,7777,7779,8000,8008,8028,80
                    80,8088,8118,8123,8180,8243,8280,8888,9090,9091,9443,9999,11371]
portvar SHELLCODE_PORTS !80
portvar ORACLE_PORTS 1024:
portvar SSH_PORTS 22
ipvar AIM_SERVERS [64.12.24.0/23,64.12.28.0/23,64.12.161.0/24,64.12.163.0/24,64.12.200.0/24,205.188.3.0/
                    24,205.188.5.0/24,205.188.7.0/24,205.188.9.0/24,205.188.153.0/24,205.188.179.0/24,205.188.248.0/
                    24]
var RULE_PATH rules
var SO_RULE_PATH so_rules
var PREPROC_RULE_PATH preproc_rules
config disable_decode_alerts
config disable_tcpopt_experimental_alerts
config disable_tcpopt_obsolete_alerts
config disable_tcpopt_ttcp_alerts
config disable_tcpopt_alerts

config checksum_mode: all
config pcre_match_limit: 3500
config pcre_match_limit_recursion: 1500
config detection: search-method ac-split search-optimize max-pattern-len 20
config event_queue: max_queue 8 log 3 order_events content_length
dynamicpreprocessor directory /usr/local/lib/snort_dynamicpreprocessor/
dynamicengine /usr/local/lib/snort_dynamicengine/libsf_engine.so
dynamicdetection directory /usr/local/lib/snort_dynamicrules
preprocessor normalize_ip4
preprocessor normalize_tcp: ips ecn stream
preprocessor normalize_icmp4
preprocessor normalize_ip6
preprocessor normalize_icmp6
preprocessor frag3_global: max_frags 65536
preprocessor frag3_engine: policy windows detect_anomalies overlap_limit 10 min_fragment_length 100 timeout 180
preprocessor stream5_global: max_tcp 8192, track_tcp yes, track_udp yes, track_icmp no max_active_responses 2 min_
                    response_seconds 5
preprocessor stream5_tcp: policy bsd, detect_anomalies, require_3whs 180, \
   overlap_limit 10, small_segments 3 bytes 150, timeout 180, \
    ports client 21 22 23 25 42 53 79 109 110 111 113 119 135 136 137 139 143 \
        161 445 513 514 587 593 691 1433 1521 2100 3306 6070 6665 6666 6667 6668 6669 \
```

Listing 5b. *Creating the /usr/local/etc/snort/snort.conf configuration file (Note: the IP range for HOME_NET is set to a standard class C subnet. Change the HOME_NET range to your specific network range). This configuration is adapted from the official snort.conf from the registered ruleset available from Sourcefire at http://www.snort.org*

```
        7000 32770 32771 32772 32773 32774 32775 32776 32777 32778 32779, \
    ports both 80 311 443 465 563 591 593 636 901 989 992 993 994 995 1220 1414 1830 2301 2381 2809 3128 3702 5250
                    6907 7001 7702 7777 7779 \
        7801 7900 7901 7902 7903 7904 7905 7906 7908 7909 7910 7911 7912 7913 7914 7915 7916 \
        7917 7918 7919 7920 8000 8008 8028 8080 8088 8118 8123 8180 8243 8280 8888 9090 9091 9443 9999 11371
preprocessor stream5_udp: timeout 180
preprocessor http_inspect: global iis_unicode_map unicode.map 1252 compress_depth 65535 decompress_depth 65535
preprocessor http_inspect_server: server default \
    chunk_length 500000 \
    server_flow_depth 0 \
    client_flow_depth 0 \
    post_depth 65495 \
       oversize_dir_length 500 \
    max_header_length 750 \
    max_headers 100 \
    ports { 80 311 591 593 901 1220 1414 1830 2301 2381 2809 3128 3702 5250 7001 7777 7779 8000 8008 8028 8080 8088
                    8118 8123 8180 8243 8280 8888 9090 9091 9443 9999 11371 } \
    non_rfc_char { 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 } \
    enable_cookie \
    extended_response_inspection \
    inspect_gzip \
    normalize_utf \
    unlimited_decompress \
    apache_whitespace no \
    ascii no \
    bare_byte no \
base36 no \
       directory no \
       double_decode no \
       iis_backslash no \
       iis_delimiter no \
       iis_unicode no \
       multi_slash no \
   utf_8 no \
       u_encode yes \
       webroot no
preprocessor rpc_decode: 111 32770 32771 32772 32773 32774 32775 32776 32777 32778 32779 no_alert_multiple_requests
                    no_alert_large_fragments no_alert_incomplete
preprocessor bo
preprocessor ftp_telnet: global inspection_type stateful encrypted_traffic no
preprocessor ftp_telnet_protocol: telnet \
    ayt_attack_thresh 20 \
    normalize ports { 23 } \
    detect_anomalies
preprocessor ftp_telnet_protocol: ftp server default \
    def_max_param_len 100 \
```

Listing 5c. *Creating the /usr/local/etc/snort/snort.conf configuration file (Note: the IP range for HOME_NET is set to a standard class C subnet. Change the HOME_NET range to your specific network range). This configuration is adapted from the official snort.conf from the registered ruleset available from Sourcefire at http://www.snort.org*

```
    ports { 21 2100 3535 } \
    telnet_cmds yes \
    ignore_telnet_erase_cmds yes \
    ftp_cmds { ABOR ACCT ADAT ALLO APPE AUTH CCC CDUP } \
    ftp_cmds { CEL CLNT CMD CONF CWD DELE ENC EPRT } \
    ftp_cmds { EPSV ESTA ESTP FEAT HELP LANG LIST LPRT } \
    ftp_cmds { LPSV MACB MAIL MDTM MIC MKD MLSD MLST } \
    ftp_cmds { MODE NLST NOOP OPTS PASS PASV PBSZ PORT } \
    ftp_cmds { PROT PWD QUIT REIN REST RETR RMD RNFR } \
    ftp_cmds { RNTO SDUP SITE SIZE SMNT STAT STOR STOU } \
    ftp_cmds { STRU SYST TEST TYPE USER XCUP XCRC XCWD } \
    ftp_cmds { XMAS XMD5 XMKD XPWD XRCP XRMD XRSQ XSEM } \
    ftp_cmds { XSEN XSHA1 XSHA256 } \
    alt_max_param_len 0 { ABOR CCC CDUP ESTA FEAT LPSV NOOP PASV PWD QUIT REIN STOU SYST XCUP XPWD } \
    alt_max_param_len 200 { ALLO APPE CMD HELP NLST RETR RNFR STOR STOU XMKD } \
    alt_max_param_len 256 { CWD RNTO } \
    alt_max_param_len 400 { PORT } \
    alt_max_param_len 512 { SIZE } \
    chk_str_fmt { ACCT ADAT ALLO APPE AUTH CEL CLNT CMD } \
    chk_str_fmt { CONF CWD DELE ENC EPRT EPSV ESTP HELP } \
    chk_str_fmt { LANG LIST LPRT MACB MAIL MDTM MIC MKD } \
    chk_str_fmt { MLSD MLST MODE NLST OPTS PASS PBSZ PORT } \
    chk_str_fmt { PROT REST RETR RMD RNFR RNTO SDUP SITE } \
    chk_str_fmt { SIZE SMNT STAT STOR STRU TEST TYPE USER } \
    chk_str_fmt { XCRC XCWD XMAS XMD5 XMKD XRCP XRMD XRSQ } \
    chk_str_fmt { XSEM XSEN XSHA1 XSHA256 } \
    cmd_validity ALLO < int [ char R int ] > \
    cmd_validity EPSV < [ { char 12 | char A char L char L } ] > \
    cmd_validity MACB < string > \
    cmd_validity MDTM < [ date nnnnnnnnnnnnnn[.n[n[n]]] ] string > \
cmd_validity MODE < char ASBCZ > \
    cmd_validity PORT < host_port > \
    cmd_validity PROT < char CSEP > \
    cmd_validity STRU < char FRPO [ string ] > \
    cmd_validity TYPE < { char AE [ char NTC ] | char I | char L [ number ] } >
preprocessor ftp_telnet_protocol: ftp client default \
    max_resp_len 256 \
    bounce yes \
    ignore_telnet_erase_cmds yes \
    telnet_cmds yes
preprocessor smtp: ports { 25 465 587 691 } \
    inspection_type stateful \
    enable_mime_decoding \
    max_mime_depth 20480 \
    normalize cmds \
    normalize_cmds { ATRN AUTH BDAT CHUNKING DATA DEBUG EHLO EMAL ESAM ESND ESOM ETRN EVFY } \
```

Listing 5d. *Creating the /usr/local/etc/snort/snort.conf configuration file (Note: the IP range for HOME_NET is set to a standard class C subnet. Change the HOME_NET range to your specific network range). This configuration is adapted from the official snort.conf from the registered ruleset available from Sourcefire at http://www.snort.org*

```
    normalize_cmds { EXPN HELO HELP IDENT MAIL NOOP ONEX QUEU QUIT RCPT RSET SAML SEND SOML } \
    normalize_cmds { STARTTLS TICK TIME TURN TURNME VERB VRFY X-ADAT X-DRCP X-ERCP X-EXCH50 } \
    normalize_cmds { X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN XLICENSE XQUE XSTA XTRN XUSR } \
    max_command_line_len 512 \
    max_header_line_len 1000 \
    max_response_line_len 512 \
    alt_max_command_line_len 260 { MAIL } \
    alt_max_command_line_len 300 { RCPT } \
    alt_max_command_line_len 500 { HELP HELO ETRN EHLO } \
    alt_max_command_line_len 255 { EXPN VRFY ATRN SIZE BDAT DEBUG EMAL ESAM ESND ESOM EVFY IDENT NOOP RSET } \
    alt_max_command_line_len 246 { SEND SAML SOML AUTH TURN ETRN DATA RSET QUIT ONEX QUEU STARTTLS TICK TIME TURNME
                    VERB X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN XLICENSE XQUE XSTA XTRN XUSR } \
    valid_cmds { ATRN AUTH BDAT CHUNKING DATA DEBUG EHLO EMAL ESAM ESND ESOM ETRN EVFY } \
    valid_cmds { EXPN HELO HELP IDENT MAIL NOOP ONEX QUEU QUIT RCPT RSET SAML SEND SOML } \
    valid_cmds { STARTTLS TICK TIME TURN TURNME VERB VRFY X-ADAT X-DRCP X-ERCP X-EXCH50 } \
    valid_cmds { X-EXPS X-LINK2STATE XADR XAUTH XCIR XEXCH50 XGEN XLICENSE XQUE XSTA XTRN XUSR } \
    xlink2state { enabled }
preprocessor ssh: server_ports { 22 } \
                 autodetect \
                 max_client_bytes 19600 \
                 max_encrypted_packets 20 \
                            max_server_version_len 100 \
                 enable_respoverflow enable_ssh1crc32 \
                 enable_srvoverflow enable_protomismatch
preprocessor dcerpc2: memcap 102400, events [co ]
preprocessor dcerpc2_server: default, policy WinXP, \
detect [smb [139,445], tcp 135, udp 135, rpc-over-http-server 593], \
    autodetect [tcp 1025:, udp 1025:, rpc-over-http-server 1025:], \
    smb_max_chain 3
preprocessor dns: ports { 53 } enable_rdata_overflow
preprocessor ssl: ports { 443 465 563 636 989 992 993 994 995 7801 7702 7900 7901 7902 7903 7904 7905 7906 6907 7908
                    7909 7910 7911 7912 7913 7914 7915 7916 7917 7918 7919 7920 }, trustservers, noinspect_encrypted
preprocessor sensitive_data: alert_threshold 25
output unified2: filename snortunified2.log, limit 128
include classification.config
include reference.config
include $RULE_PATH/local.rules
include $RULE_PATH/snort.rules
include threshold.conf
```

Listing 6. *Downloading and installing barnyard2*

```
cd /usr/src/snort
wget http://www.securixlive.com/download/barnyard2/
                    barnyard2-1.9.tar.gz
tar -xzf barnyard2-1.9*
cd barnyard2-1.9
./configure --with-mysql
make
make install
```

Listing 7. *Creating the /usr/local/etc/barnyard2.conf (Note: Use your hostname and primary interface for the config options hostname and interface)*

```
config reference_file:      /usr/local/etc/snort/reference.config
config classification_file: /usr/local/etc/snort/
                    classification.config
config gen_file:             /usr/local/etc/snort/gen-msg.map
config sid_file:                   /usr/local/etc/snort/
                    sid-msg.map
config hostname:        snorbyHost
config interface:       em0
input unified2
output database: log, mysql, user=snorby
                    password=s3cr3tsauce dbname=snorby
                    host=localhost
```

Listing 8. *Downloading and install PulledPork. (Note: In this example, a made-up oink code of a576b9aac7622306349ee5a6f7ebabce is used. Also each sed command should be on single line.)*

```
cd /usr/src/snort
/usr/local/bin/wget http://pulledpork.googlecode.com/
                    files/pulledpork-0.6.1.tar.gz
tar -xzf pulledpork*
cd /usr/src/snort/pulledpork-0.6.1
sed -IBAK -e 's/|<oinkcode>/|a576b9aac762230634
                    9ee5a6f7ebabce/g' /usr/src/
                    snort/pulledpork-0.6.1/etc/
                    pulledpork.conf
sed -IBAK -e "s/rule_url=https:\/\/rules.emerging
                    threats.net\/|/#/g" /usr/src/
                    snort/pulledpork-0.6.1/etc/
                    pulledpork.conf
rm -rf /usr/src/snort/pulledpork-0.6.1/etc/
                    pulledpork.confBAK
/usr/src/snort/pulledpork-0.6.1/pulledpork.pl -c /usr/
                    src/snort/pulledpork-0.6.1/etc/
                    pulledpork.conf
```

Barnyard2

http://www.securixlive.com/barnyard2/. An important configuration option used in `snort.conf` is the output plugin. Because the job of an IDS/IPS is to inspect packets off the wire as fast as possible, the sensor can not spare resources to handle the sending of event data. This is the function of barnyard2, which takes the unified2 output file from Snort and processes the events as they come in. The unified2 output allows Snort to quickly log event data to a binary file-format which barnyard2 can process separately. Instructions for downloading and compiling barnyard are shown in Listing 6.

Barnyard contains a configuration file with several examples of different output options. For the case of Snorby, we are sending the Snort event data to a MySQL database on the local server. Listing 7 details the configuration file setup for barnyard2.

Listing 9. *Downloading and installing Snorby and configuring MySQL*

```
cd /usr/ports/devel/git
make -DBATCH install clean

/usr/local/bin/gem install prawn --no-rdoc --no-ri
/usr/local/bin/gem install rake -v 0.8.7 --no-rdoc --
                    no-ri
/usr/local/bin/gem install rails --no-rdoc --no-ri
/usr/local/bin/gem install mysql --no-rdoc --no-ri
/usr/local/bin/gem install passenger --no-rdoc --no-ri
/usr/local/bin/passenger-install-apache2-module -a

cd /usr/src/snort/
wget --no-check-certificate https://github.com/Snorby/
                    snorby/zipball/2.2.6
tar -xzf 2.2.6
mv Snorby-* Snorby
mv /usr/src/snort/Snorby /usr/local/www/Snorby

/usr/local/etc/rc.d/mysql-server onestart
mysqladmin -u root password 's3cr3tsauce'

chown -R www:www /usr/local/www/Snorby
cd /usr/local/www/Snorby
bundle pack
bundle install --path vender/cache
rake snorby:setup
mysql -uroot -ps3cr3tsauce -e "GRANT ALL ON snorby.*
TO snorby@localhost IDENTIFIED BY 's3cr3tsauce';"
mysql -uroot -ps3cr3tsauce -e "FLUSH PRIVILEGES;"
chown -R www:www /usr/local/www/Snorby
```

Pulled Pork

Listing 8 details the process of downloading and configuring Pulled Pork to work with the Snort ruleset.

http://pulledpork.googlecode.com. In order to handle the new binary rules and simplify rule management in Snort, J.J. Cummings created Pulled Pork, which automatically downloads and configures Snort rules. Pulled Pork can be configured to download multiple rulesets including the Emerging Threats Open and Pro rulesets. In this example of Pulled Pork, only the registered Snort ruleset is used. An oinkcode must be retrieved by registering for free at *http://www.snort.org*.

Listing 9 describes the next step of installing and configuring Snorby. The next step is to install and configure Snorby. There are several prerequisites that are

Listing 10. *Configuration of Apache to work with Snorby (Note: each sed command is on one line. Also, Servername is set to 192.168.1.155 in this example. The actual value should be the IP used to configure the interface)*

```
sed -IBAK -e 's/"\/usr\/local\/www\/apache22\/data"/"\
                    /usr\/local\/www\/Snorby\/public"/
                    g' /usr/local/etc/apache22/
                    httpd.conf
sed -IBAK -e 's/#Include etc\/apache22\/extra\/httpd-
                    vhosts.conf/Include etc\/apache22\
                    /extra\/httpd-vhosts.conf/g' /usr/
                    local/etc/apache22/httpd.conf
rm -rf /usr/local/etc/apache22/httpd.confBAK

echo "LoadModule passenger_module  /usr/local/
                    lib/ruby/gems/1.8/gems/
                    passenger-3.0.7/ext/apache2/
                    mod_passenger.so" >> /usr/local/
                    etc/apache22/httpd.conf
echo "PassengerRoot  /usr/local/lib/ruby/gems/1.8/
                    gems/passenger-3.0.7" >> /usr/
                    local/etc/apache22/httpd.conf
echo "PassengerRuby /usr/local/bin/ruby18" >> /usr/
                    local/etc/apache22/httpd.conf
echo "" >> /usr/local/etc/apache22/httpd.conf
echo "ServerTokens Prod" >> /usr/local/etc/apache22/
                    httpd.conf
echo "ServerName  192.168.1.155:80" >> /usr/local/etc/
                    apache22/httpd.conf

cat << EOF > /usr/local/etc/apache22/extra/httpd-
                    vhosts.conf
<VirtualHost *:80>
 ServerName 192.168.1.155
DocumentRoot /usr/local/www/Snorby/public
 <Directory "/usr/local/www/Snorby/public">
 AllowOverride all
 Order deny,allow
 Allow from all
 Options -MultiViews
 </Directory>
</VirtualHost>
EOF
```

Listing 11. *Script to create snorbyfix.sh, which fixes the delayed_job issue, and the cronjob to run it*

```
cat << EOF > /usr/local/etc/snorbyfix.sh;
#!/bin/sh
#Local fixes for Snorby with Apache
#
TEST=\'ps aux|grep delayed_job\';

if [ ! \$TEST ];
then
        cd /usr/local/www/Snorby;
        /usr/local/bin/ruby script/delayed_job start;
fi
EOF

chmod 700 /usr/local/etc/snorbyfix.sh;

cat << EOF >> /etc/crontab;
#
#Cronjob for snorby fix.
*/5      *       *       *       *       root    /usr/
                    local/etc/snorbyfix.sh
EOF
```

Listing 12. *Configuration of HTTP Accept Filter module, and local additions for rc.conf.local to start the services. (Note: 192.168.1.155 and snorbyHost are just example IP and hostnames to use. Make sure these settings match your configuration.)*

```
echo 'accf_http_load="YES"' >> /boot/loader.conf
echo "192.168.1.155    snorbyHost
                    snorbyHost.localdomain" >> /etc/
                    hosts

echo 'mysql_enable="YES"' >> /etc/rc.conf.local
echo 'apache22_enable="YES"' >> /etc/rc.conf.local
echo 'snort_enable="YES"' >> /etc/rc.conf.local
echo 'barnyard2_enable="YES"' >> /etc/rc.conf.local
```

installed with RubyGems, the package manager for Ruby. The passenger module is also configured and MySQL is setup to accept the Snorby database.

Once Snorby has been setup and configured in MySQL, Apache needs to be configured to launch Snorby as an embedded Ruby application via the passenger module. Listing 10 shows the steps required to configure Apache to work with Snroby. There was a minor issue with one of the scripts in Snorby that will be addressed in a later release. The problem is that the `delayed_job` Ruby script does not run

Listing 13. *Creating the snort and barnyard2 rc scripts. (Note: In this example, the interface is set to em0, as this enables the rc script to manage the processes via their pid. Change this to the correct interface name if using a different network card (in this case, not using an Intel e1000 driver)*

```
cat << EOF > /usr/local/etc/rc.d/snort;
#!/bin/sh


#
# PROVIDE: snort
# REQUIRE: LOGIN
# KEYWORD: shutdown
#
# Add the following lines to /etc/rc.conf.local or
                    /etc/rc.conf
# to enable this service:
#
# snort_enable (bool):   Set to NO by default.
#               Set it to YES to enable snort.
# snort_config (path):   Set to /usr/local/etc/snort/
                    snort.conf
#               by default.
#

. /etc/rc.subr

name="snort"
rcvar=\${name}_enable

command=/usr/local/bin/\${name}
pidfile=/var/run/\${name}_em0.pid

load_rc_config \$name

: \${snort_enable="NO"}
: \${snort_config="/usr/local/etc/snort/snort.conf"}

command_args="--pid-path \$pidfile -c \$snort_config -D
                    not arp"

run_rc_command "\$1"

EOF
chmod 555 /usr/local/etc/rc.d/snort;
```

```
cat << EOF > /usr/local/etc/rc.d/barnyard2;
#!/bin/sh

# \$FreeBSD\$
#
# PROVIDE: barnyard2
# REQUIRE: mysql LOGIN
# KEYWORD: shutdown
#
# Add the following lines to /etc/rc.conf.local or
                    /etc/rc.conf
# to enable this service:
#
# barnyard2_enable (bool):   Set to NO by default.
#               Set it to YES to enable barnyard2.
# barnyard2_config (path):   Set to /usr/local/etc/
                    barnyard2.conf
#               by default.
#

. /etc/rc.subr

name="barnyard2"
rcvar=\${name}_enable

command=/usr/local/bin/\${name}
pidfile=/var/run/\${name}_em0.pid

load_rc_config \$name

: \${barnyard2_enable="NO"}
: \${barnyard2_config="/usr/local/etc/barnyard2.conf"}

command_args="-c \$barnyard2_config -d /var/log/snort
-f snortunified2.log -w /var/log/snort/barnyard2.waldo
                    --pid-path \$pidfile -D"

run_rc_command "\$1"

EOF
chmod 555 /usr/local/etc/rc.d/barnyard2;
```

in the background. Listing 11 provides a fix for this issue by using a cronjob to make sure the script is running and if not will start it. In order for Apache to startup correctly, the HTTP Accept Filter module must be loaded. Adding this to `/boot/loader.conf` will cause the module to be loaded at boot time. In addition, several rc scripts are created in Listing 12 for Snort and Barnyard to streamline the starting of the server.

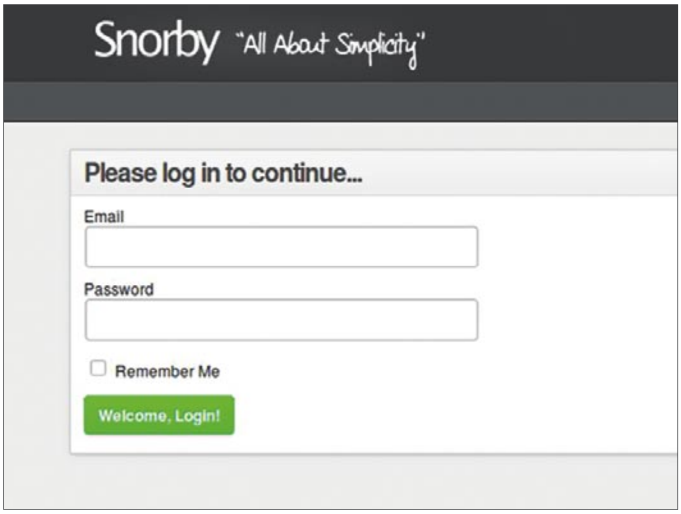


Figure 1. *Snorby login screen*

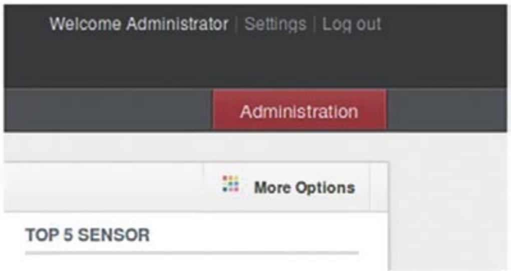


Figure 2. *Administration Menu*

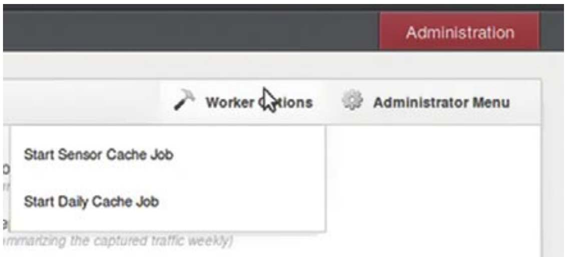


Figure 3. *Starting jobs*

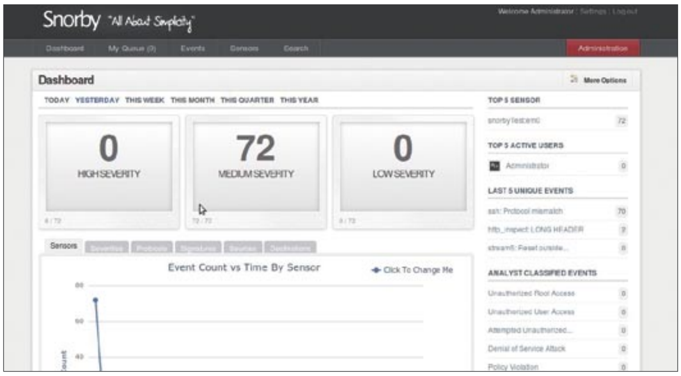


Figure 4. *Snorby Dashboard*

References

- Automatic Snorby Installer for FreeBSD: *http://rootedyour.com/enhanced/snorbyInstall.sh, http://www.shirkdog.us/snorbyInstall.sh*
- FREEBSD-INSTALL: *http://www.freebsd.org/doc/handbook/install-start.html*
- Snort: *http://www.snort.org*
- Snorby: *http://www.snorby.org*
- Snorby on FreeBSD install: *http://global-security.blogspot.com/2009/07/snorby-for-snort-recipe-with-barnyard2.html*
- Pulled Pork: *http://pulledpork.googlecode.com/*
- Additional Snorby Information: *http://www.corelan.be/index.php/2011/02/27/cheat-sheet-installing-snorby-2-2-with-apache2-and-suricata-with-barnyard2-on-ubuntu-10-x/*

Listing 13 creates the necessary snort and barnyard2 rc scripts to be placed in `/usr/local/etc/rc.d`. With the local settings in `/etc/rc.conf.local`, snort and barnyard will automatically start at boot.

At this point, reboot the FreeBSD server and once it starts back up, Snorby should be accessible. In our example, the URL to use in the browser would be *http://192.168.1.155*.

The email to use as the username is *snorby@snorby.org* with the password snorby. After logging in, click on the Administration menu shown in Figure 2.

Inside the Administration Menu, click Worker Options and click *Start Sensor Cache Job* and *Start Daily Cache Job* as shown in Figure 3.

Once the jobs have been started, Snorby is ready to use. Navigate to the dashboard, or click on *Events* to see the most current IDS events.

Conclusion

This article only scratches the surface of configuring a production ready IDS system. The focus of the examples given is to familiarize the readers with the requirements of how to get the system up and running on FreeBSD. In order to make this process even simpler, I have created a shell script that automates the entire process which is available from the references under *Automatic Snorby Installer for FreeBSD*.

MICHAEL SHIRK

Michael Shirk is a BSD zealot who has worked with OpenBSD and FreeBSD for over 6 years. He works in the security community and supports Open-Source security products that run on BSD operating systems. He wishes to thank Thomas Conway and J.J. Cummings for testing the instructions in this article.

Full Disk Encryption on FreeBSD

On systems (for instance laptop computers) that may be physicaly accessed or stolen by untrusted persons, encrypting sensitive pieces of data should be mandatory.

What you will learn…

- Installing a full encrypted FreeBSD

What you should know…

- Basics of FreeBSD installation process

Encrypting only the data partitions may be weak since it may allow to retrieve a piece of data via cached informations on the encrypted partitions and is vulnerable to any attacker who is able to install a rootkit on the encrypted file-system. This article will explain how to fully encrypt a FreeBSD system.

Theory

FreeBSD supports two encryption technologies: GDBE and GELI. Since only GELI allows to boot the encrypted paritions, we will only use GELI for the purpose of this article. Unfortunately, sysinstall does not support creation of encrypted partitions. In order to install FreeBSD on encrypted partitions, we will have to create these partitions without sysinstall. In order to solve this problem, we will perform a two step installation. We will first install our system on a small unencrypted partition and after that we will migrate this install (excepting the boot loader to encrypted partitions)

First install

This is an almost standard FreeBSD install. During partitionning, you will only create a small / partition (512 MB should be enough) and do not set any mount points for the other partitions (that will be encrypted soon). In order to create a partition without setting a mount point with sysinstall, you may create the partition with a dummy mount point and then remove it using *m*. It may be painful to use multiple encrypted partitions since this means multiple passphrases to provide at each reboot. For this article, we will only use one encrypted partition (`/dev/da0s2d` while our small unencrypted partition will be `/dev/da0s1a`).

Since the / partition is really small, you will have to choose the minimal distribution set. This is really not a problem, since you will be able to add other distribution sets after the encryption process. In order to facilitate the encryption process, you may install the net/rsync port. With only 512 MB available, the smartest way is to use the pre-compiled package.

Encrypting the OS

As recommended by the official GELI documentation you may use a key file associated with a passphrase.

```
# dd if=/dev/random of=/boot/geli.key bs=64 count=1
# geli init -s 4096 -K /boot/geli.key /dev/da0s2d
```

With these two commands, you will create the key file and encrypt the partition (geli init will ask for a passphrase). If you use non-US keyboard layout, be careful with special characters in the passphrase. During the boot process, the passphrase will be asked with a US keymap loaded and your non-US keymap will be set up a few steps after.

Listing 1. *Retrieving and Installing rsync*

```
# ftp -a ftp.fr.freebsd.org
Connected to ftp.free.org.
220 Welcome to free.org FTP service.
331 Please specify the password.
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd pub/FreeBSD/ports/packages/net/
250 Directory successfully changed.
ftp> get rsync-3.0.8.tbz
local: rsync-3.0.8.tbz remote: rsync-3.0.8.tbz
229 Entering Extended Passive Mode (|||17682|).
150 Opening BINARY mode data connection for rsync-3.0.8.tbz (263860 bytes).
100% |****************************************************************|   257 KB    9.07 MB/s    00:00 ETA
226 Transfer complete.
263860 bytes received in 00:00 (8.72 MB/s)
ftp> ^D
221 Goodbye.
[root@majinbox ~]# pkg_add rsync-3.0.8.tbz
```

The following command will deblock the encrypted partition by creating /dev/da0s2d.eli

```
# geli attach -k /boot/geli.key /dev/da0s2d
```

So, in order to format the partition, you will have to issue the following command:

```
# newfs /dev/da0s2d.eli
```

Now, you are able to mount the partition and synchronise / (excepting /boot)

```
# mount /dev/da0s2d.eli /mnt
# rsync -av --progress --exclude /boot --exclude /mnt / /mnt/
```

Booting the encrypted OS

In order to boot using the encrypted partition, you first have to prepare the current / to become /boot:

```
# mkdir /mnt/boot
# vi /mnt/etc/fstab
```

The content of /mnt/etc/fstab should look like this:

```
# Device          Mountpoint   FStype  Options  Dump    Pass#
/dev/da0s1a       /boot        ufs     rw       1       1
/dev/da0s2d.el    /            ufs     rw       2       2
```

After that, you will have to modify the /boot/loader.conf in order to tell kernel that it needs to use GELI (with the right key file) and mount / from GELI. You simply need to add the following lines to /boot/loader.conf:

```
geom_eli_load="YES"
geli_ad0s1d_keyfile0_load="YES"
geli_ad0s1d_keyfile0_type="da0s2d:geli_keyfile0"
geli_ad0s1d_keyfile0_name="/boot/geli.key"
vfs.root.mountfrom="ufs:da0s2d.eli"
```

Once you reboot, you will be asked for a passphrase and use the encrypted partition. It will be necessary to clean up old / stuff from the /boot.

Possible improvements

Since the /boot cannot be encrypted, it may be backdoored by what is called an evil maid attack (*http://theinvisiblethings.blogspot.com/2009/10/evil-maid-goes-after-truecrypt.html*). One solution will be to put it on a removable media (for instance a memory-card or an USB key) and store this media apart from the encrypted computer.

MATTHIEU BOUTHORS

Matthieu Bouthors is a French *NIX enthusiast for a decade now. Working for a French hosting company, he aims find open source solutions meeting the high-level requirements of its customers.

What It Takes

Starting and Running an Open Source Certification Program, Part II

Last month, in the first article in this series, we discussed the People aspect of running an Open Source certification program such as the BSD Certification Group (BSDCG). We discussed the types of people you'll need in your program- SMEs, Writers, Translators, Technical Experts, Managers, the Advisory Group, and your Psychometrician.

This month we'll look into the *Processes* that are involved in setting up a certification program. A certification program should be set up as a business- not just an informal group of people. If you are going to charge money for your exam, you will need some sort of organization to take money in, pay money out, and pay your taxes- hopefully more of the former than the latter two.

Setting Up the Business

There is always a flurry of activity in getting an Open Source project started, and a certification program is no exception. Almost immediately people will be asking how to pay for the certification. Unless you plan on stuffing all the money in your mattress, you'll want to set up a bank account. But a bank is going to want to know if you are setting up a personal account or a business account. So the first thing you'll want to decide is how to get set up as a business.

There's a stigma against *business* in the Open Source world which is unfortunate because when it's done well, a business actually helps an Open Source project. A business provides a focal point for legal, financial, and organizational issues. A random group of people has a more difficult time explaining their organizational bona fides to government agencies, financial institutions, and companies looking to do business with your certification program.

The technical details of setting up a business can be challenging. There are several types of business entities such as a for-profit corporation, non-profit corporation, sole proprietorship, S-corporation, *Limited Liability Partnership* (LLP) and more. And we're only discussing US organizations here. Other countries have different types of legal business entities, and you'll need specialized expertise in these cases.

From the beginning, the BSDCG really wanted to help the BSD community. Our focus was not so much on making money as it was on getting BSD systems more widely adopted. We decided to incorporate as a US non-profit organization. The actual mechanics of doing that were fairly straightforward.

If this is the way your group wants to go, you will need to do the following:

- Elect a group of people to act as *officers* (CEO, VP, Treasurer) for your certification organization. Any business, even a non-profit corporation, needs officers- a group of people authorized to speak and act for the organization. Note that the US *Internal Revenue Service* (IRS) requires that a majority of incorporating officers must be US citizens. In our case, we started with just 3 individuals, and 2 of the 3 were US citizens.
- Your group will have to draft *Articles of Incorporation* that identify you as a US non-profit corporation in a US state. It's handy, but not required, to live in the state where you register. (Delaware is often chosen for its business friendly environment, even though the business is not physically located there. If you do that, you'll need to contract with an *agent* that will forward all legal correspondence to your physical location.) This is a formal step and should be reviewed by a lawyer and an accountant if possible. Depending on the state involved there may be an initial registration cost, and yearly renewals.

- Your group will need to draft and approve *Bylaws* – published rules that determine how the company will operate, how decisions are made, who is authorized to handle money, deal with legal issues, vote on choosing a Board of Directors, vote on organizational issues, keep rules of order for meetings, and other mundane issues. There are many examples of corporate Bylaws to pick and choose from, but take care- your business will be bound to follow these rules once they are adopted, so choose wisely. As with most corporate issues, it's helpful to have a lawyer review the Bylaws.
- Once you are a registered business, you'll need to get an *Employer Identification Number* (EIN) from the IRS. You need this even if you don't have any employees. It's a number the IRS, banks, financial services companies, and other business entities use to identify your business. At a minimum, you'll need it for filing your yearly tax return.
- In the US, once you decide to incorporate as a non-profit, you also have to decide what kind of non-profit you want to be. Because there are significant tax advantages for donors, most non-profit organizations want to be recognized by the IRS as a *501(c)(3)* charity organization- one that the IRS allows tax deductions for donors. This is another formal step with a lengthy application process, and it will cost money. Most of the effort is in documenting your certification organization, its business plan, sources of funding, website, recent activities, public benefits, etc. If you are somehow fortunate to gain 501(c)(3) status, you will have to eventually pass the *Public Support Test*- at least 1/3 of your total support (basically income and donations) normally comes from from the general public with corporate donors or sponsors subjected to certain limits although there are some exceptions.

  It turns out that the IRS uses a number of well established precedents to evaluate new non-profit companies wishing to be a 501(c)(3) charity organization. The general idea is that the organization must benefit the public as a whole, not just a subset of like-minded individuals. One precedent is for a medical certification organization that looks very similar to the BSDCG. The IRS determined that organization is really a 501(c)(6) organization, known as a *Business League* (described as an association of persons having some common business interest), for which donations are not tax deductible. Since that was an established precedent, they applied it to the BSDCG. While they listened politely while we tried to convince them otherwise, they remained convinced that we were a *business league*.
- Once you have your EIN, set up a business bank account. You're now officially open for business. Banks will charge monthly services fees to keep your account. If you're fortunate enough to keep a hefty average daily balance, they will generally waive those fees, but be alert- they will often try to sell you extra services which you probably don't need.

It's a very good idea to engage someone with bookkeeping experience for non-profits to keep track of your finances.

You can do it yourself, but the tax rules for non-profits differ from those of for-profit companies and it's worth getting someone who knows what they are doing. There are also commercial software packages that handle non-profit organizations. These will need updating periodically, as tax laws change frequently, even for non-profits.

Setting Up the Certification

Getting the certification underway involves many separate, interdependent tasks, a lot of which will depend on your wider community. In our case, the BSD community has been involved from the beginning in deciding what kind and number of certifications should be offered, what they should be named, what knowledge domains they should consider, and what kind of test delivery method should be available. In the beginning, the BSDCG set up mailing lists to give everyone a chance to express opinions (some more strongly than others) and exchange ideas. We also set up a live discussion channel, `#bsdcert` on *freenode.net*.

There were so many contrasting viewpoints and differences of opinion on so many levels, it was very hard to get rolling. Eventually, someone piped up and said that we needed to do a *Job Task Analysis* (JTA) because this was needed to identify the specific skills a BSD system administrator needs on the job. It was a requirement for a psychometrically valid exam. This was the first time many of us had heard anything about psychometrics and was a key turning point.

A JTA identifies all the important tasks that are regularly performed by the position being certified (in our case a BSD system administrator). It also identifies how often those tasks are performed. These tasks are then organized into *domains of knowledge* that are expected to be known by candidates seeking certification. In our case, someone stepped up and helped us put together an online survey for the first Job Task Analysis for

BSD system administrators. We had several thousand respondents.

Our psychometrician helped us refine the results into our core knowledge domains which were published as the official BSD Associate (BSDA) exam objectives document (*http://www.bsdcertification.org/downloads/pr_20051005_certreq_bsda_en_en.pdf*).

A meeting of *Subject Matter Experts* (SMEs) was scheduled and a large number of questions (items) were developed and reviewed at the meeting. The psychometrician gave directions on how to score items for acceptability. The result was a large collection of pretty good items for the BSDA exam. The next task was to select which items from the pool were to be included on the initial exam form (collection of items).

We held these exam items in a standalone database not accessible from the Internet. The reason was simple- if that host was compromised, and the items published on the Net, all our work would have been in vain.

We were now ready to test the exam.

Testing the Exam

The next step was a Beta Test. We needed to know if the questions were too easy, too hard, not appropriate, or not understandable for non-native English speakers. We selected 100 beta testers (the minimum recommended by our psychometrician) and held a Beta exam. By offering a reduced price for the production (real) exam, we were able to get many Beta testers. This was a significant plus for us.

We decided that the Beta exam would be distributed by the paper-and-pencil exam method. The exam was delivered, and the results tabulated. During this time, our psychometrician pored over the results and gave us extensive assistance on selecting the correct items for the official draft of the exam, as well as an analysis of the computations for the *cut score* i.e. the minimum passing score.

Other Considerations

If you've gotten this far, you are about to launch the official exam, but there are still several items you need to consider. You still need to set a price for the exam, figure out how to accept money via credit card, PayPal, checks, or even cash; arrange one or more delivery methods, and figure out some way to communicate to the candidate whether they passed or failed. We'll discuss certificates and delivery methods below. First, it's time to consider the costs.

Hopefully, you've already discussed how much to charge for the exam within your group. If so, you've heard all the complaints about exorbitant fees for exams. It's true that some certification exam fees are way out of line and are designed only to line the pockets of the certification company.

So before you set a price for your certification exam, consider the following:

- You'll need to recover all your costs to at least break even. Factor in everything you have spent money on so far. Include capital expenses (*CapEx*) like computers to host your item database, certificates, printers for certificates, seals, etc. CapEx items are tangible- things you can hold, view, or use. Operating expenses (*OpEx*) are generally service related expenses as discussed in the next bullet.
- Consider your ongoing OpEx items such as startup costs including registration, legal, and accounting fees. Include costs for web hosting and maintenace fees. Also include the delivery costs (see below), such as exam printing and shipping. Shipping can be a large expense over time if you are shipping exams packs, and certificates overseas. Don't forget advertising- you'll want people to know about your exam. Advertising in industry trade magazines is one effective way to do that, but there are other, lower cost methods such as user groups, conference talks, and blogs.
- Ongoing psychometric costs (OpEx). This can be substantial if you intend to maintain the certification over time. Each certification program you run should be annually reviewed to determine if any items need to be added or subtracted. If (when) you meet with your psychometrician in person, allow for reimbursement of travel costs. They are unlikely to travel on their own dime.
- If you raise money in some other way than exam sales, such as setting up a corporate sponsorship program, selling training materials, CDs, DVDs, books, etc., include that money in your calculations.

You'll also want to look into your crystal ball to see how many candidates will take your exam. You'll need to estimate wisely here, as overestimating exam income can result in very strained financial situations.

Finally, figure out your total income and expenses, factor in a modest *fudge factor*, and divide by the number of exam candidates you expect to definitely pay for and take the exam in one year. The result is your initial exam price. If it seems too high (or too low) recheck your figures and assumptions. Whatever it is, remember that this is your initial price, and will likely change as your program matures.

The BSDCG went through the exact process above. Our goal was to produce an exam that is affordable by anyone regardless of where they are from. We met that goal by setting our initial exam price at $75 USD in 2008.

It's time to consider creating your certificates. Consider the certificate carefully- a certificate will be visible to the successful candidate (and everyone else) for several years. You want to project success and professionalism, so you should design an attractive looking certificate out of high quality materials. There are many examples to choose from. We chose a simple, elegant design for the BSDA certificate. We also designed a 1-color embossed gold seal for the certificate that is applied individually to each certificate. And unlike many certification organizations, the signatures on a BSDA certificate are real on every single one.

Now for the delivery question- how will you actually deliver the exam? You've spent a long time developing a high-quality, psychometrically valid exam. If you just put it up on a web page with a *Submit* button, you will lose all of that advantage. Why? Because there is no way to positively identify the actual test taker with an online exam over the Internet. In the certification world, just like the academic world, people will pay their friends to take the exam for them if they think it will help them pass. The only reliable way to ensure that the exam candidate is who they say they are is with a proctor. An exam proctor checks each candidate's identification, and personally administers the exam to ensure there is no cheating.

Arranging to deliver proctored exams can be done in several ways:

- Do it yourself at industry conferences, user groups, and corporate training events. The advantage is that you have total control over the testing environment. The disadvantage is that you can only reach a small number of people at a time, and travel costs can be a concern.
- Arrange for your exam to be given by a trusted proctor at user groups, corporate training events, or industry conferences. This is a good method if you can find reliable, honest proctors. The BSDCG uses this method and chooses proctors from the computer industry that already have an established reputation. Advantages- good control; disadvantages- still only reaching a small number of candidates.
- Engage a commercial test delivery company such as Vue or Prometric. Advantages- this can greatly expand your exam availability; disadvantages- can be very expensive (over $50,000 per year) with some companies.

It's worth looking at more than just the two big names. The BSDCG engaged with Schroeder Measurement Technologies (SMT *http://www.smttest.com/*) for a fraction of the above cost and we've now expanded our delivery offerings to over 300 test centers in some 20 countries. The price for the computer based exam had to change to accommodate this arrangement, but even at $150 USD, the exam is still affordable to almost everyone. The paper-and-pencil exam is still just $75 USD.

An ideal proctor is an individual with no knowledge of the exam subject material. This means they couldn't answer any questions on the exam even if they wanted to. While it's not always possible to find an ideal proctor, you can ensure a positive exam experience by drafting a proctor guidelines document that discusses what the proctor should do in setting up, delivering, wrapping up an exam session, and returning all the materials. Also, all proctors should sign a Non-Disclosure Agreement, and agree to keep the contents of the exam materials private and safe at all times.

Setting Up for the Final Exam Launch

Finally, you are ready to launch. Now is the time to engage your media team to get the word out about your new exam. Tune up that advertising you've already been doing to announce your new certification. Get the word out with blogs, user groups, social media and any other way you can.

Once it's offically released, you'll need to score the exam and communicate the results. We'll discuss these items in next month's article on the Technology elements involved in your certification program.

Governance

Before you sit back and celebrate, you still have some work to do. For the certification program to be successful, you have to have regular Board of Directors meetings to discuss any issues. You need to periodically review the exam results with the psychometrician. You should also regularly review the finances to see whether you are making or losing money. You also need to regularly review contracts with your psychometrician and test delivery companies. And don't forget to update your website.

But governance of your program is more than just a list of activities. Since you've put a lot into the program and you want it to be the best it can be, consider looking into accreditation of your certification program by the American National Standards Institute (ANSI *http://ansi.org/*).

They've developed an accreditation program based on the ANSI/ISO/IEC 17024 (*https://www.ansica.org/wwwversion2/outside/PERgeneral.asp?menuID=2*) international standard. The accreditation recognizes certification organizations that maintain best practices including the above practices and more.

To achieve this accreditation, your certification program joins the ANSI program and *starts the clock* on the accreditation. You are assigned two ANSI auditors- one who does the initial audit and one for the final audit.

The initial audit is where you find out all your governance issues – documentation, management reviews, procedures, etc. that need corrective action. Once you make progress you schedule the second audit where they take an even closer look to make sure your program is fully compliant with ANSI/ISO/IEC 17024. At the end of eighteen months, you will either have succeeded or been placed on remediation track where you have even more work to do to improve your program.

More information on ANSI accreditation and the ANSI/ISO/IEC 17024 standard can be found on their website. It is expensive, but the benefits are worth it – you get access to international and government markets and you get recognized as a best in class certification program. Get that and you'll really have something to celebrate.

****

In next month's article, we'll finish up by discussing the Technology aspects of a certification program- surveys, collaboration, test construction, delivery, scoring and reporting, and security. We'll also look at key metrics that can be useful in the ongoing governance and administration of the program.

JIM BROWN

Jim Brown has worked in the computer industry with continuous Unix involvement in development or administration since the early 1980s. His experience includes applications, systems and database programming, in a variety of languages. One of the founders of the BSD Certification Group, he is helping to develop the BSD Professional certification. He currently lives in Northwest Arkansas, USA.

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

Interview with Paul Schenkeveld

After working with many UNIX and UNIX-like operating systems since 1983, Paul Schenkeveld was asked to maintain the FreeBSD section of the NLUUG (NetherLands Unix User Group) FTP mirror in the early 1990s. This was even before FreeBSD 1.0 came out. Of course he downloaded FreeBSD to see what it is and that's how it all begun. Since release 2.2, FreeBSD was powerful enough to fulfill real tasks in his network, the first one was a firewall and now Paul maintains more than 100 FreeBSD machines and several hundred virtual hosts running FreeBSD and still has enough time to do projects and visit conferences.

Paul has participated in all nine EuroBSDcon editions so far and visited BSDcan and AsiaBSDcon several times. During EuroBSDcon 2009, Paul and his fellow BSD fan Cor Hilbrink were asked to help another friend to bring EuroBSDcon (back) to The Netherlands in 2011. Unfortunately the other friend had to drop out due to time constraints but they managed to get four other people to join the organising comittee and four others to join the program comittee. Although it takes a lot of time to make the conference happen, Paul says he would never have missed this experience, and that it is amazing how many people support the initiative and help out with advice or otherwise when asked.

Q&A:

BSD magazine: For those of our readers who are not familiar with it – can you tell us what is your conference about? What is your mission?

Paul Schenkeveld: EuroBSDcon brings together developers and users of various operating systems based on the famous BSD operating system which was derived from UNIX in the 1970s at the Computer Systems Research Group of the Berkeley University in California. The most well known representatives are NetBSD, OpenBSD and FreeBSD but the conference is also open to members of the DragonFly BSD and Darwin (open source part of Apple OS-X) communities.

The conference is preceded by two days of tutorials covering various topics related to BSD operating systems and a developer conference, where many active developers gather to discuss development issues face to face.

The conference itself features many talks, some very technical, others more user-experience oriented, some explaining brand-new or even future additions, others reflecting on the history of BSD or commonly used software that runs on BSD. Except for one or two plenary sessions, most talks will be offered in two or three parallel track so at every time you'll most likely find at least one talk that interests you.

During the coffee and tea breaks, lunches and the social event on saturday evening there is plenty of time to socialize, find long sought answers to burning questions about your favorite BSD, exchange experiences and make new friends or visit the exhibitors presenting BSD related material.

The ultimate mission of these conferences is to strengthen the BSD community in a very friendly atmosphere and together with the sister conferences like AsiaBSDcon and BSDcan bring those who drive these communities together on a regular base.

BSDmag: You mentioned AsiaBSDcon and BSDcan – do you cooperate with organisers of those conferences in any way?

Paul: We do cooperate with the organizers of AsiaBSDcon, in fact Hiroki Sato who is the driving force behind AsiaBSDcon has been very helpful by giving some inside information about their conference. We have visited BSDcan too, met Dan Langille who organizes that conference and looked how BSDcan is organized but so far we have not had a direct reason to contact him. We have contacted people who organized EuroBSDcon in Germany, Switzerland, France and the United Kingdom with specific questions and they have all been very helpful.

Finally, we got help from people who organized or are organizing other conferences like SANE, Fosdem and hacker camps like HAR and ETH0.

The power of open source projects, like the BSD operating systems, lies in the communities that drive the projects. It really feels like having a very big, friendly and helpfull family.

BSDmag: Let us know who should attend to the conference?

Paul: If you do anything with one of the BSD operating systems, think about doing so or use another operating system but feel it's time to broaden your horizon, these BSD conferences are really the place to be.

BSDmag: So someone new and inexperienced with BSD will find something for himself as well?

Paul: That's our goal. Looking at this years proposals, there are many interesting talks for people new to BSD.

BSDmag: How big the conference will be – in terms of attendees and speakers? What are your expectations?

Paul: It's always difficult to tell how many people will actually register but looking at previous editions of EuroBSDcon we expect between 150 and 250 attendees.

BSDmag: How did EuroBSDcon start and how has it evolved to where it is today?

Paul: EuroBSDcon started as an experiment by Nik Clayton and Paul Richards in 2001 in Brighton, UK. There had been BSD conferences in northern America before but they thought that a BSD conference in Europe would be a good complement. In 2002, some of my friends here in the Netherlands were inspired and organized a second edition in Amsterdam. The audience really showed their interest in a sequel to these first two conferences so we saw seven more conferences in Germany (twice), Switzerland, Italy, Denmark France and England. The format was always similar (and similar to the yearly conferences in Ottawa, Canada, and Tokyo, Japan) but the European conferences have been one-offs (except for the two conferences in Germany) organized by a fresh team every year again so each organization team had to re-invent the wheel.

During the conference in 2010 we set up a meeting with the organizers of 2010 and 2011, someone who will most probable organize 2012 and members of the FreeBSD, OpenBSD and NetBSD communities to discuss the future of EuroBSDcon. We agreed that we need to form a Foundation that will govern this years and future conferences so we can hand over the experience from one team to the next. We also decided that we should have someone from previous years organization and someone from next years organization in our team organizing the conference this year to carry over experiences and we started to document the whole process for future teams so they can concentrate on making new mistakes instead of repeating old mistakes.

BSDmag: Can you tell us more about this foundation?

Paul: Work to get the new foundation up and running is currently stalled because everybody is very busy at the moment but the idea is that this foundation will collect and keep records from this years EuroBSDcon organization and passes this knowledge and experience on to next years organizing comittee. Another benefit of having this foundation is that financial resources can be transferred from one conference to another, now if one conference has some money left at the end it cannot easily be transferred to next years conference to cover some potential loss.

BSDmag: Do you have any special plans for the anniversary? Will it be different from previous Conferences, and how?

Paul: We have some ideas but it is too early to talk about it.

BSDmag: It's tenth EuroBSD conference. Have european BSD community evolved a lot since the first one?

Paul: The spirit seems very much the same back then and now, EuroBSDcon has always been and remained an event where all people interested in BSD operating systems feel at ease whether they are (kernel) developer or beginner. Of coourse topics have shifted over the years and fortunately there are enough new faces every year so we really don't have to worry about becoming extinct soon :-)

BSDmag: Is there any chance you could reveal who will be speaking at the EuroBSD Con 2011?

Paul: Since the program committee is still reviewing proposals and has not yet decided on the final list of talks and tutorials that will be accepted, it is too early to reveal names. However, around the time this issue of BSD Magazine sees the daylight a preliminary schedule of tutorials and talks should be available at *www.eurobsdcon.org*.

As usual there will be a number of well known speakers who are always a guarantee for an interesting tutorial or talk but on our proposals list I've seen also quite a number of new names with abstracts for tutorials and talks never presented before that promise a very interesting mix of subjects.

BSDmag: What kind of criteria do you use to select the topics for the conference agenda?

Paul: Setting up a program for a conference like EuroBSDcon is a very complex puzzle so there are many answers to this question. Let me try to explain a bit.

First of all we depend on people submitting proposals for a talk or tutorial. It takes a lot of advertising to relevant mailing lists and web pages but also private emails to people that we know who would be able to submit a worthful talk or tutorial to elicit enough proposals to choose from, unfortunately speakers are not lining up to talk these days.

Then there is always the question of money, the entrance fees of all participants is really not sufficient to pay for travel and hotel expenses for the number of speakers you'd like to invite but we are happy that there are still some companies and organizations that are willing to sponsor conferences like ours. But the budget limits the number of speakers we can actually invite.

Then there is the issue of diversity in the BSD world, there are three major, well known BSD projects NetBSD, OpenBSD and FreeBSD, and there are Darwin, the open source part of Apples MacOS, Dragonfly BSD and some projects derived from the three major BSDs. Ideally you'd like to have a balanced mix of talks from all the relevant projects. So we have a program committee with members from the three major BSD families to make sure we do not lean over to just one or two sides.

Besides EuroBSDcon, there are several sister conferences like AsiaBSDcon every march and BSDcan every may. We look at their list of talks to make sure we are not a repetition of their schedule because of the overlap in audience. There are not enough submissions to make a complete new program with only talks that have not been presented before so we try to overlap only with talks that are high-profile enough to repeat. Besides that the members of the program committee each search in the community they are most familiar with to find people who have done innovative work that has not yet been presented and try to lure them to submit a talk.

The submission period for this years EuroBSDcon closed may 30 and the program committee members are now evaluating the submissions to form an interesting lineup of talks, invited speakers have to be notified by end of june so the speakers can send in their final papers in time and we can start all practical arrangements like travel, hotel rooms.

BSDmag: Do you allow commercial presentations at EuroBSDcon? And if so – how many companies will attend?

Paul: As long as the content of the presentation is well related to BSD and we deem the presentation interesting enough for our audience, I see no harm in a commercial presentation but the focus of the conference is always on technical content so a commercial presentation of a very technical nature makes much more chance than a sales talk with little or no technical content.

BSDmag: How do you see the future of BSD and EuroBSD conference?

Paul: BSD has been around for about three and a half decades and looking at the evolution of the BSD families that we know today I think that BSD has a very bright future. The user base seems to grow steadily over the years and there is an amazing number of bright people contributing to the various BSD projects. Being projects that are driven by many committed volunteers and not by one or more companies that have to pay mega salaries and bonuses to their managers and have to keep their shareholders happy too make these projects very healthy and future-proof.

It's amazing to see how many brilliant young people have found their way into these projects, initially contributing under supervision by experienced committers but independently once they have shown to have the right set of mind and deliver good quality code. I happen to know some of these people and see how happy they are with the recognition they get for their work. Being part of one of the BSD families seems to boost your career more than any job at a commercial company ever could.

BSD Magazine Team

EuroBSDcon

2011

The **Anniversary**

BSD Daemon © Marshall Kirk McKusick. Used with permission. http://www.mckusick.com/copyright.html

6 until 9 October, 2011

Meeting Plaza, **Maarssen**

Address: **Planetenbaan** 100
3606 AK Maarssen
The Netherlands

GPS: **N**52.12840, **E**5.0360

10th European BSD Conference

http://2011.eurobsdcon.org/