MAGAZINE

BSD

FOR NOVICE AND ADVANCED USERS

LINUX JAILS IN PC-BSD

INSIDE

VOL.6 NO.12
ISSUE 12/2012(41)
1898-9144

FreeNAS® 8 POWERED

TrueNAS™

UNIFIED. SCALABLE. FLEXIBLE.

Across all industries the demands of data infrastructure have soared to new heights.

As capacity requirements continue to rise at an ever-increasing rate, performance must not be compromised. The hybrid architecture and advanced software capabilities of the TrueNAS appliance enable users to be more agile, effectively manage the explosion of unstructured data and deploy a centralized information storage infrastructure. Whether it's backing virtual machines, business applications, or web services, there's a TrueNAS appliance suited to the task.

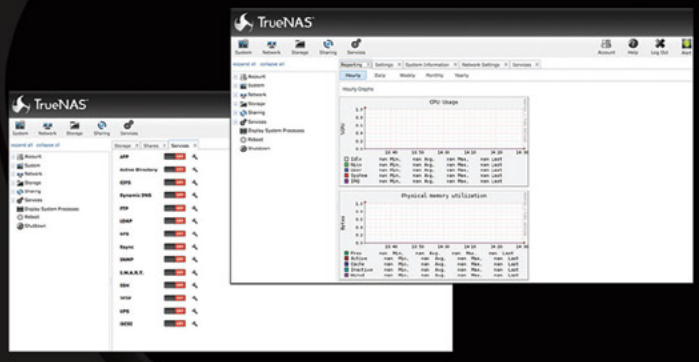
TrueNAS™ Storage Appliances: Harness The Cloud

iXsystems' TrueNAS Appliances offer scalable high-throughput, low latency storage

All TrueNAS Storage Appliances feature the Intel® Xeon® Processors 5600 series, powering the fastest data transfer speeds and lowest latency possible. TrueNAS appliances come in three lines: Performance, Archiver, & High Availability. High-performance, high-capacity ioMemory modules from Fusion-io are available in the TrueNAS Enterprise, Ultimate, and Archiver Pro models.

Key Features:

- **One or Two Six-Core Intel® Xeon® Processors 5600 series**
- **Share Data over CIFS, NFS and iSCSI**
- **Hybrid storage pool increases performance and decreases energy footprint**
- **128-bit ZFS file system with up to triple parity software RAID**



*Optional component

	TrueNAS Pro	TrueNAS Enterprise	TrueNAS Ultimate	TrueNAS Fileshare	TrueNAS Archiver Pro	TrueNAS Pro-HA	TrueNAS Enterprise-HA	TrueNAS Ultimate-HA
	PERFORMANCE			**ARCHIVER**		**HIGH AVAILABILITY**		
Fusion-io Card		X	X		X			
Deduplication					X			
High Availability						X	X	X
Gigabit NICs	Quad	Dual	Dual	Dual	Dual	Six	Quad	Dual
10 Gigabit NICs		Dual*	Quad*		Dual*			Dual
Max Main Memory	48Gb	96Gb	192Gb	48Gb	192Gb	48Gb	96Gb	192Gb
Max Capacity	220TB	500TB	1.5PB	580TB	2.2PB	250TB	310TB	1.4PB
Rack Units	2U	2U/4U	4U	2U	4U	3U	3U	Dual 3U

iXsystems®

Call iXsystems toll free or visit our website today!

1-855-GREP-4-IX | www.iXsystems.com

intel Xeon® inside™

Dear Readers,

We present you the last issue of 2012. We open it with a new column, where Rob Somerville describes in light-hearted and amusing way the battles he fights as a system administrator. It's not technical article, but rather a rant. However, time to time it helps to know that others experience the same difficulties as we do. This new column aim is to create a discussion over some small, but important matters.

Our cover article is about installing and configuring Linux jails in PC-BSD. Its author Patrick Allen wrote a really nice tutorial, so you can spend one of those cold winter evenings on checking out his tips and experimenting with jails.

The network administrators might be interested in setting up and managing a small business DNS/DHCP server, which you will find in admin section. Where Antonio Gentile described his own work experiences.

In December issue you will also see the continuations of three popular series, written by well-known contributors – Michael Shirk (TrustedBSD), Rob Sommerville (Apache) and Luca Ferrari (PostgreSQL). All three series will soon reach their culmination. BSD beta testers were asked to share their ideas about new topics for series. If there is any particular topic that you think could be our magazine's bestseller – write us.

On the last few pages are two overviews presenting three events concerning BSD. Those who didn't managed to attend any of them, have an opportunity to see, what they missed.

Since it's Christmas time, together with our fellow editors from Hakin9 and PenTest Magazines, we prepared for you some presents! More you can find out in this issue from Hakin9 Magazine ad, so don't miss it!

Patrycja Przybyłowicz
Editor of BSD Magazine
& BSD Team

MAGAZINE BSD

Editor in Chief:
Ewa Dudzic
ewa.dudzic@software.com.pl

Supportive Editor
Patrycja Przybyłowicz
patrycja.przybylowicz@software.com.pl

Contributing:
Rob Somerville, Kris Moore, Luca Ferrari, Antonio Francesco Gentile, Patrick Allen, Michael Dexter

Top Betatesters & Proofreaders:
Barry Grumbine, Bjørn Michelsen, Paul McMath, Imad Soltani, Luca Ferrari, Cleiton Alves, Eric Geissinger, Mani Kanth, Zander Hill, Ahmed Aneeth, Norman Golisz, Rob Cabrera, Will Clayton

Special Thanks:
Denise Ebery

Art Director:
Ireneusz Pogroszewski

DTP:
Ireneusz Pogroszewski
ireneusz.pogroszewski@software.com.pl

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

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

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

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

Advertising Sales:
Patrycja Przybyłowicz
patrycja.przybylowicz@software.com.pl

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

Software 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.

Let's Talk

How To

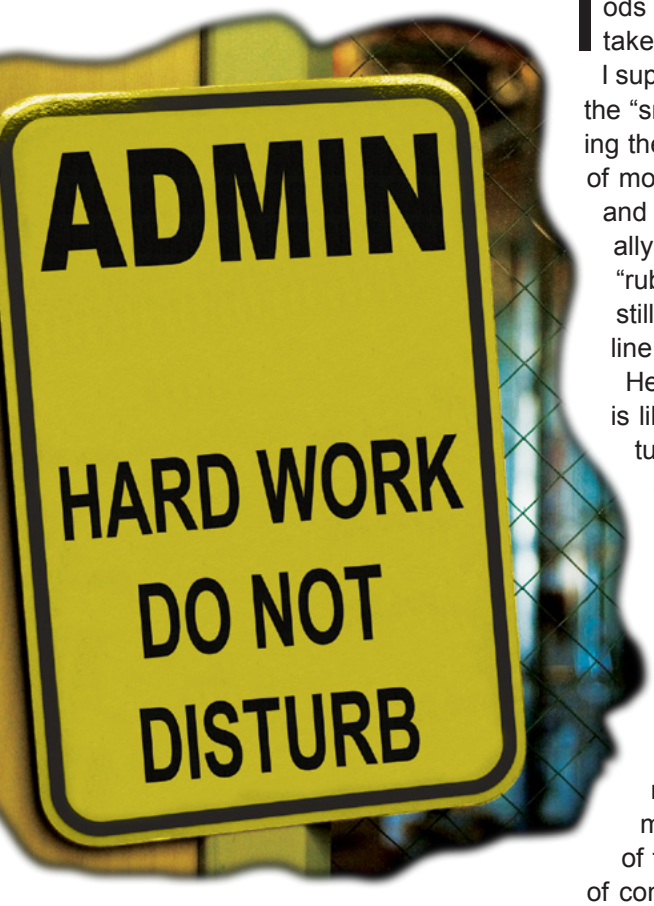
Admin

Security

Overview

The Sandbox

8:45 Monday morning. I fill the espresso filter basket with a good measure of Italian coffee, flick the switch to espresso, and 60 seconds later am rewarded with a demitasse of viscous caffeine, complete with the requisite creamy head. Coffee is an essential part of the I.T. toolkit, especially when deadlines loom and the disconnect between customer, 3rd party supplier and the gap between expectations and reality becomes wider by the day.



I start the week anticipating a diet of fire-fighting, cultural clashes and periods of patiently explaining this, while we can do the impossible, miracles take a little longer.

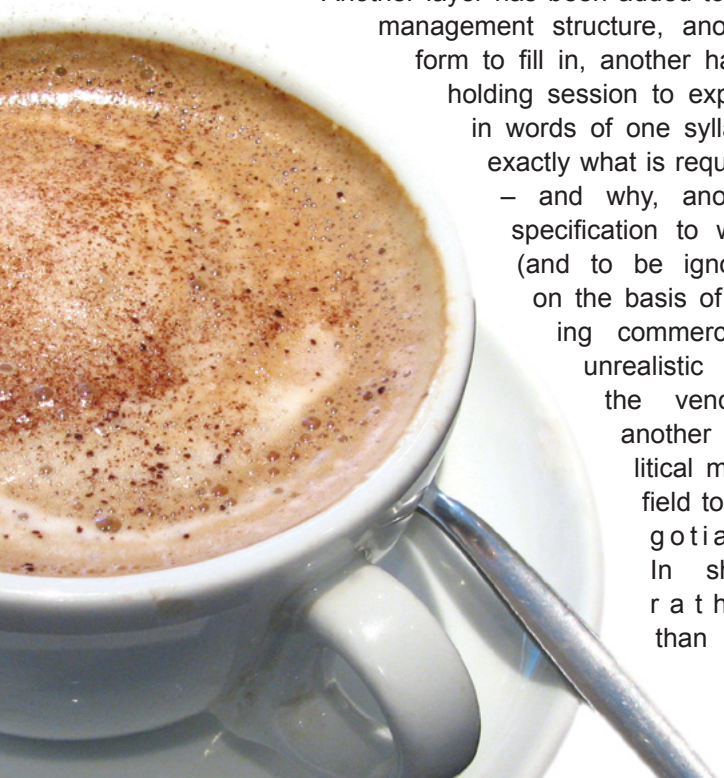
I suppose the biggest curse of the technology sector is hype – unfortunately the "smoke and mirrors" brigade always seem to have the edge in persuading the masses that technology is easy, close to infallible, and for X amount of money all your problems will be solved, and you will be a better person and a more efficient organisation as well. Sometimes corporate cultures really excel in shooting themselves in the foot – hiring external consultants to "rubber stamp" strategic decisions that are not run past I.T. first, or worse still, delivered to them as a fait accompli with an impossible deadline to match.

Here starts the beginning of the disconnect. Technology is like a plant – it needs to be rooted in good soil, nurtured and given the correct environment. Support structures need to be in place, weed killer employed, and sometimes to get the best from the plant some serious pruning is required.

Pests need to be controlled, symbiotic relationships formed, and hopefully the ecosystem will be beneficial for the plant to flourish, thrive, and bear continuous fruit. In reality, sometimes the environment is harsh, short-cuts taken, essential maintenance ignored, critical investment postponed and it is only a matter of time before the fire-fighting gets out of control and a major systems failure is experienced. Sometimes it is technology, but more often than not it is down to "expectations manage-

ment". While the new system creeps past the line of "fit for purpose", everyone knows it could have been so much better, more innovative, better engineered, future proofed. Designed by committee, any I.T. project is doomed to failure unless everyone is on board and adheres to the central vision. In reality, this is rare unless there is a benevolent dictator to steer the process. This is where the hype does the most insidious damage – the commercial realities of the vendor are to make a profit, to become an indispensable part of the customers' ecosystem, while syphoning the last vestiges of innovation and creativity through restrictive licensing or security, intellectual property rights, and a "Yes we can do it but at a price" mentality. In this scenario, the vendor becomes the dictator, and the organisation is no longer in control.

Once your most valuable resource (I.T.) is outside the doors of your organisation, you lose a crucial weapon in business – the ability to respond flexibly. Another layer has been added to the management structure, another form to fill in, another hand-holding session to explain in words of one syllable exactly what is required – and why, another specification to write (and to be ignored on the basis of being commercially unrealistic for the vendor), another political minefield to negotiate. In short, rather than getting on with the job (designing, improving and developing systems), we are turning into project managers and vendor liaison officers, while watching efficiency slide and the corresponding shrinkage of morale and job satisfaction. "Can Do" is replaced with "Not my responsibility" while the bitter spectre of how much company XYZ actually charged to modify and test 10 lines of code lies buried in some invoice in the finance department. Roll up, roll up – all of these "benefits" can be yours too, if you outsource today.



Fortunately, organisations are beginning to realise the folly of outsourcing. For all the rhetoric, good management boils down to one thing – control. Shrinking budgets are forcing companies to re-evaluate what value they get from suppliers, and the tide is turning towards Open Source and BSD unlike ever before. Smart businesses are building in-house teams, developing corporate loyalty, retaining staff and revolutionising their software platforms. Margaret Mead said "Never doubt that a small group of thoughtful, committed, citizens can change the world." I say, give me half a dozen in-house BSD guru's, the freedom to think "outside the box", and we will transform your organisation. Austerity – sometimes – can work in your favour.

ROB SOMERVILLE

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

Installing and Configuring

Linux Jails in PC-BSD

The Warden jail management tool has been redesigned for the upcoming release of PC-BSD 9.1. Many new features have been added, but one of the most exciting is the ability to create Linux jails.

What you will learn…

- Using Warden to create a Linux jail
- Configuring nat for your jail
- Installing Linux packages in the jail

What you should know…

- Basic jail and *nix concepts

Whether they prefer the CLI or a GUI, one thing most people can agree on is that The Warden is a great tool for managing jails. The Warden has been available as an add-on in PC-BSD since version 8, and is available as a port in FreeBSD as well. It now comes built-in to version 9.1 of PC-BSD and TrueOS (a variant of PC-BSD included in the install DVD that consists of FreeBSD and enhanced command line versions of PC-BSD tools).

Background

Jails are a very useful feature and a strong selling point of FreeBSD and derivative BSDs. OpenBSD, Linux and other U*ix operating systems typically use chroot to create 'safe' environments. In chroot environments, processes are confined to a particular part of the host file system, and are not allowed to access files outside of it. Therefore, if a service running in a chroot is compromised, the host system should be safe from the attack.

Jails take this a step further. In addition to the file system, jails virtualize other resources such as system users, running processes, the networking subsystem and more. Each jail even has its own root user. Jails do not provide a completely virtualized environment, as each jail sharing the host's kernel.

Linux jails offer an interesting alternative for BSD users who wish to create a virtualized environment. Some users might wish to run services in a Linux environment but don't have the resources, or the desire, to maintain a separate Linux server. While this was possible in the past with a fair amount of effort, the new version of The Warden makes this easy, and in a very secure way.

A thing to keep in mind about Linux jails however is that they are not actually utilizing a Linux kernel, so running services that require specific Linux kernel functions are not possible. This also means that if you need to make kernel modifications in order to run a particular program, these will need to be made to the host, BSD, kernel.

Preparation

Before we create our jail, there are a few things we need to take care of in order to allow it to access the internet and give us the ability to install packages. There are various ways to handle networking for jails, but for this example we will be using a loopback device, which we will call lo1. By creating this cloned interface we are giving the jail its own virtual network adapter which we can then configure separately from the actual physical adapter. For our jail in this example we will be using the address 10.0.0.1 and will only be configuring IPv4.

The first thing we need to do is create and configure the loopback device at the command line:

```
# ifconfig lo1 create
# ifconfig lo1 10.0.0.2 netmask 255.255.255.255
```

To make this persistent, add the following to `/etc/rc.conf`:

```
cloned_interfaces="lo1"
ifconfig_lo1="inet 10.0.0.2 netmask 255.255.255.255"
```

Next, we will add rules to `/etc/pf.conf` to allow the jail to use nat by mapping it to out external interface (our default ethernet adapter, `em0`). Assuming you are using the default `/etc/pf.conf` configuration file, the beginning of our file will now look like Figure 1.

The lines we added are:

- `ext_if="em0"` – A macro for our external interface so that if we switch interfaces in the future, all references in the file to that interface will not need to be changed individually,
- `jail_if="lo1"` – A macro for our loopback device that we just created,
- `jail_ip="10.0.0.0/24"` – A macro for our jail ip addresses. This enables us to create more jails using the ip range of `10.0.0.0-10.0.0.24` which will then also use the nat that we are setting up,
- nat pass on `$ext_if` from `$jail_ip` to any -> `$ext_if` – Here we are configuring pf to nat all jail traffic.

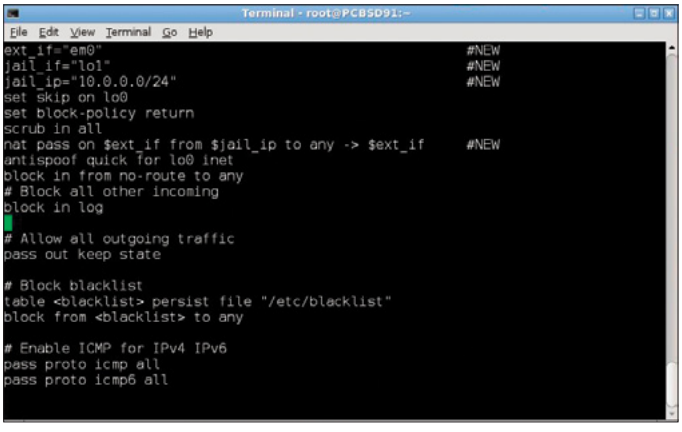


Figure 1. *Our pf.conf file after modification*

Figure 2. *Warden configuration*

We also must assure that ip forwarding is enabled for IPv4:

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

To make this persistent, add the following to `/etc/sysctl.conf`:

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

Figure 3. *IP and Hostname configuration*

Figure 4. *Jail type selection*

Lastly, we need to reload the `pf` rules:

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

Installing the jail

To begin creating our jail, start The Warden. For this example we will be using the GUI, which can be started from the PC-BSD Control Panel or from the CLI using *pc-su warden gui*. The first time you start The Warden, it will ask you to set the configuration. If you are using ethernet, the Jail Network Interface should default to `em0`, and we can use the default Jail and Temp Directory (Figure 2).

To add a new jail, click the green plus button. This will start the New Jail Wizard. The first screen asks for the IP address and hostname of our new jail. We will use the address we configured for our lo1 interface (Figure 3).

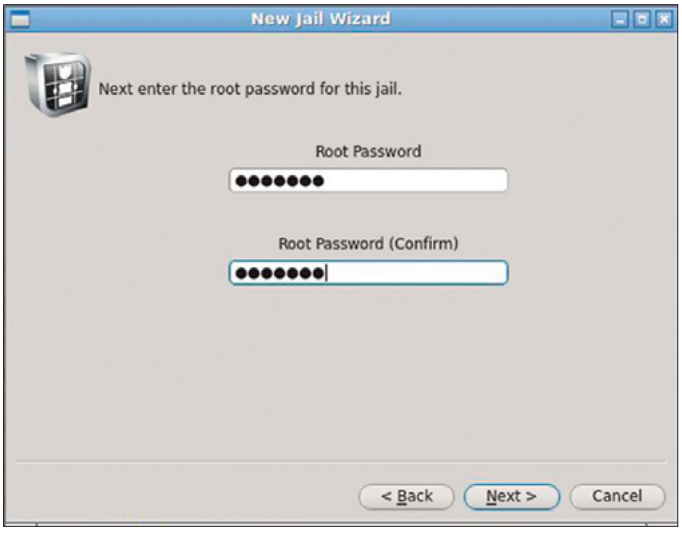


Figure 5. *Entering the root password*

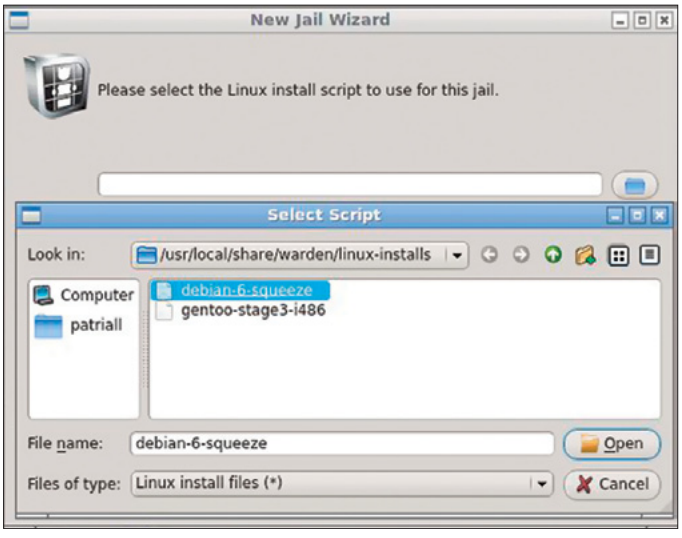


Figure 6. *Selecting an install script*

On the next screen, we are asked which type of jail we would like to create. Select Linux Jail (Figure 4).

We are then asked to enter a root password for the jail (Figure 5).

The next screen asks us to select a Linux install script for building the jail. As of PC-BSD 9.1 RC2, there are two install scripts included, debian-6-squeeze and gentoo-stage3-i486. At release more may be included, or you can easily build your own for other distros. For this example we will be using Debian (Figure 6).

Last, we are asked if we'd like to start the jail at system bootup. Make your selection and click Finish. The Warden

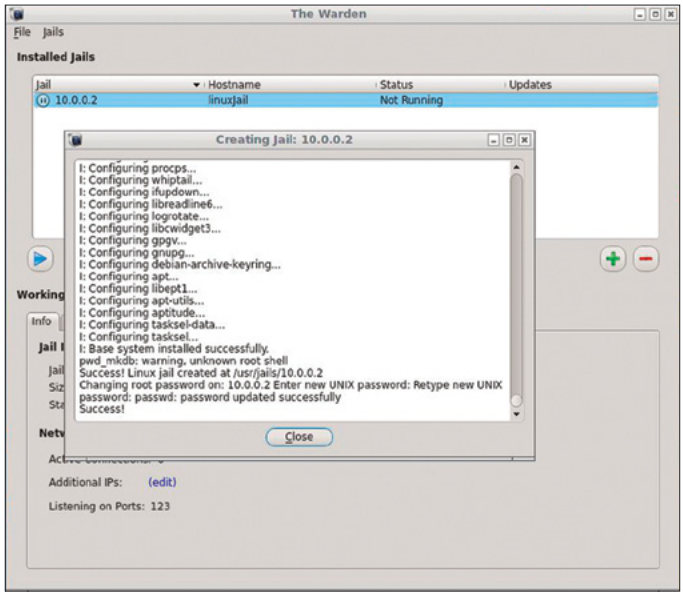


Figure 7. *Jail creation is complete*

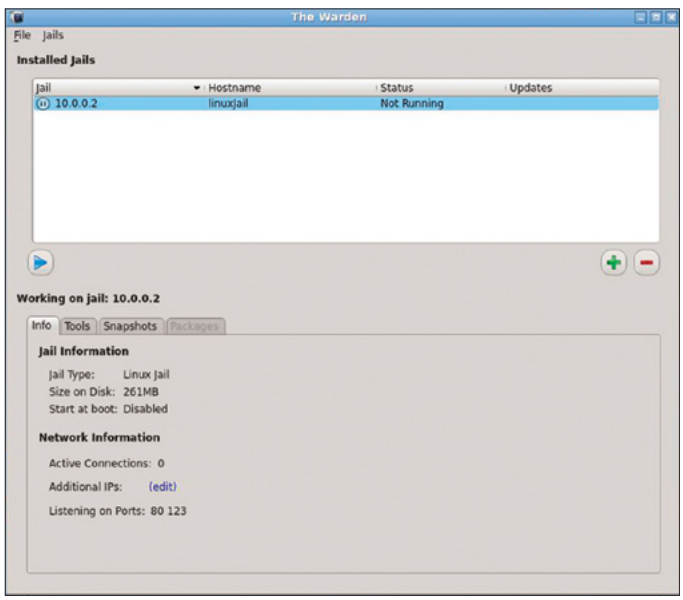


Figure 8. *Our list of jails*

then creates our jail. When it is finished, we can close the window (Figure 7).

Our jail is now created but not started. Start it by either right-clicking on the jail and selecting Start This Jail or by clicking on the blue Play button or at the terminal by issuing the command `warden start 10.0.0.2` (remember to use the name of the jail instead of the hostname).

Once the jail is started, let's look at the Tools tab. Here, we can launch a terminal to your jail, or use the Export Jail feature to create an export that you can import as a new jail at a later point.

On the Snapshots tab, if you are using ZFS for your host, you can create, restore or mount ZFS snapshots of your jail. You can also schedule daily or even hourly snapshots, as well as select how many days worth of scheduled snapshots to keep (Figure 9).

You will notice that with a Linux jail, the packages tab is grayed out. Warden does not (at least at this time) provide the ability to install Linux packages from the GUI, so we will need to choose Launch Terminal from the Tools tab and do it ourselves.

Different Linux distros come with various package management tools. For instance, Gentoo uses `emerge`, Red Hat based distros use `rpm` and `yum`, and so on. The package management tools provided with a Debian Linux jail are `apt-get` and `dpkg`. Some of the most popular Linux distros are based on Debian, so there is much documentation available on using these tools. As an example we will install thttpd, the tiny/turbo/throttling HTTP server, using apt-get.

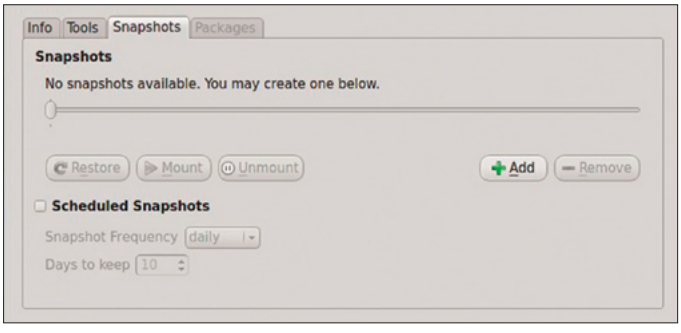


Figure 9. *Snapshot tab*

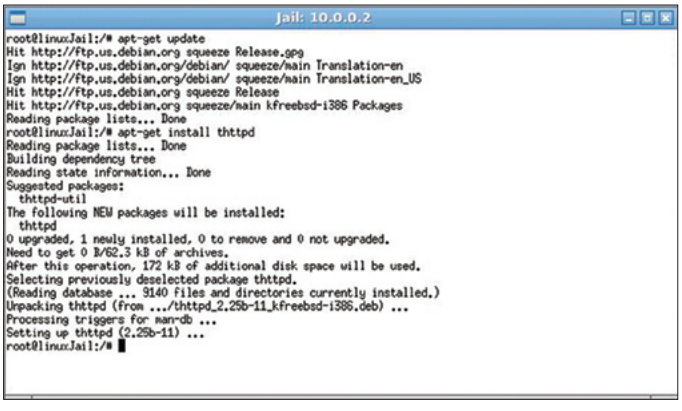


Figure 10. *Using apt-get in a Linux jail*

At the command line in your jail terminal, we will first run `apt-get update`, which makes sure our package source lists are up to date. Next, to do the install we enter `apt-get install thttpd`. If we are installing a package with dependencies, `apt-get` will ask us if we'd like to install those dependencies before installing the package. You now have a Linux jail with the thttpd server installed.

Conclusion

Jails are a great tool for system administrators, giving them the portability of being able to easily copy or move the whole environment, and the security and stability of being isolated from the host system. They are a smart alternative to traditional virtualization since they do not have the overhead of hardware emulation, providing a lightweight environment when performance is a priority. The addition of Linux adds to the usefulness and flexibility of jails, giving administrators more options than ever to set up just the type of environment they need. The information and tools explained in this article should give a user the ability to hit the ground running with Linux jails when PC-BSD 9.1, and the new version of Warden, are available. To read more about the new features in the upcoming version, visit the preview version of the new Warden documentation at *http://wiki.pcbsd.org/index.php/Warden*.

PATRICK ALLEN

Patrick Allen is a developer and DBA from Colorado Springs, Colorado. He uses AIX and Linux at work, and *BSD for pleasure. However, he still misses his first true love, his Commodore 64.

FreeBSD Enterprise

Search with Apache Solr (Part 4)

In the final part of our series on Apache Solr we will look at Apache Tika and demonstrate how to import and index document content with Apache Solr.

What you will learn…

- How to set up Apache Tika and integrate it with Solr

What you should know…

- BSD administration skills, FreeBSD Apache Solr Parts 1, 2 and 3 articles

So far, we have used Solr to access and index content found in web pages, XML files, databases and external websites. But as far as using Solr in the enterprise is concerned, how can we access disparate documents such as PDF and Microsoft Word files? This is where Apache Tika is invaluable – supporting over 14 different types of document formats (Table 1 – Tika supported document formats).

Processing the files takes place in two stages. In stage one, Tika reads the file then parses and extracts the relevant meta-data. In the second stage the extracted data is posted to Solr. Searching the file comprises two stages, the query is sent to Solr by the user and Solr returns the link to the document. The user can then view or download the document via their browser (Figure 1 – Parsing, posting and querying). In the case of documents (PDF, DOC etc.) both the content and the meta-data is extracted. In the case of media files, archives etc, only the meta-data is extracted, for example exif data in the case of images, and MP3 tags in the case of MP3 files.

Physical access to the documents can be achieved in a number of ways. In the case of a content management system, the file or attachment is uploaded via the CMS interface, and then processed and stored on the web-server. Accessing files stored en masse on a file-server could be accomplished by using fusefs-ntfs and Apache. Irrespective of method used, Tika must be able to read, parse and post to Solr, and conversely Solr must be able to point back to the file either via a physical share (on an intranet for example) or via an HTTP link to the file.

Table 1. *Tika supported document formats*

Format	Supports
Content and metadata extraction	
HyperText Markup Language	Virtually any kind of HTML
XML and derived formats	XHTML, OOXML and ODF
Microsoft Office document formats	OLE 2 Compound Document and Office Open XML including DOC, XLS, PPT,MPP etc.
OpenDocument Format	ODF
Portable Document Format	PDF
Electronic Publication Format	EPUB
Rich Text Format	RTF
Text formats	TXT, CSV
Metadata	
Compression and packaging formats	bzip2, tar and zip
Audio formats	MP3, FLAC
Image formats	JPG,PNG.GIF
Video formats	FLV, MP4
Java class files and archives	JAR
The mbox format	MBOX

Required Files

If you have been following the series from the beginning, tika-core-1.1.jar and tika-parsers-1.1.jar should already be in the `collectionX/lib` directory, with tika-app-1.2.jar in the `tmp/solr` directory. If not, Tika will have to be manually downloaded and compiled using Maven. See (Table 2/3). You will also need some sample files to import – in this example I will use the previous Solr 3 article in various file formats.

Step 1. Configure Tika

Log in to your test Solr server, stop your running Solr instance, and then create a new collection with the extensive schema from collection1:

```
# su
# /usr/local/etc/rc.d/tomcat7 stop
# cd /home/solr
# cp -R collection3 collection4
```

Creating the new collection

Edit solr.xml to reflect the new collection by adding the following lines to the `<cores>` section (Listing 1).

```
# vi solr.xml
```

Editing solr.xml (Listing 1). Change the cores line to read (Listing 2). Remove the line (Listing 3), and replace it with (Listing 4). Flush the index data:

```
# rm collection4/data/index/*
# rm collection4/data/tlog/*
```

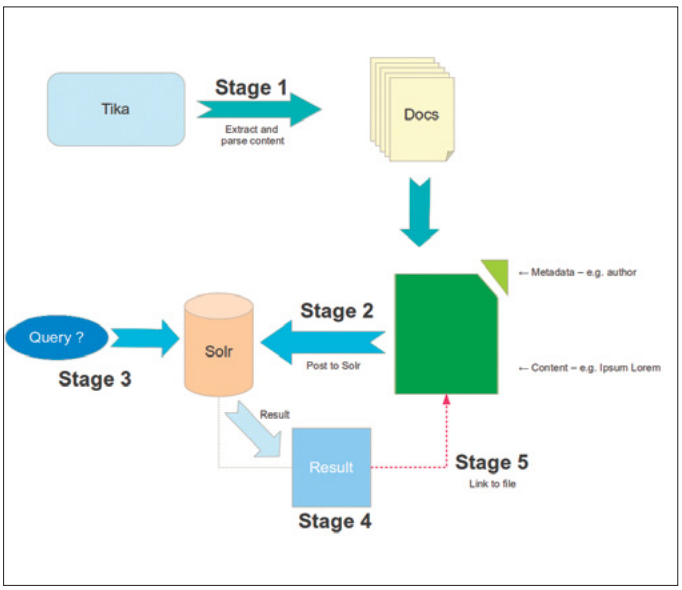


Figure 1. *Parsing, posting and querying*

Flushing the indexes

Ensure that file and group rights are correct:

```
# chown -R www:www collection4
# ls -alh collection4/lib/tika*
```

Updating the user rights and checking Tika JAR

You should see 2 Tika JAR files present (Figure 2). Do not copy the tika-app jar in `/tmp/solr` to your lib directory – this will cause the indexing to fail.

We could use Tomcat to serve the files, but Apache is better for this. Install and configure it to point to the `/exampledocs` directory, and then restart Tomcat:

```
# pkg_add -r apache22
# echo 'apache22_enable="YES"' >> /etc/rc.conf
# rm -fr /usr/local/www/apache22/data/
# ln -s /home/solr/exampledocs /usr/local/www/apache22/
                  data
```

Listing 1. *XML*

```
<core schema="schema.xml"
   instanceDir="/home/solr/collection4/"
   name="collection4"
   config="solrconfig.xml"
   dataDir="/home/solr/collection4/data"
/>
```

Listing 2. *XML*

```
<cores  adminPath="/admin/cores" zkClientTimeout="${zk
                  ClientTimeout:15000}" >
```

Listing 3. *XML*

```
 <core instanceDir="collection1/" name="collection1"/>
```

Listing 4. *XML*

```
<core schema="schema.xml"
   instanceDir="/home/solr/collection1/"
   name="collection1"
   config="solrconfig.xml"
   dataDir="/home/solr/collection1/data"
/>
```

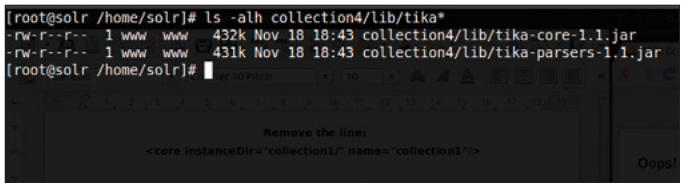


Figure 2. *Tika JAR files*

```
# /usr/local/etc/rc.d/apache22 restart
# /usr/local/etc/rc.d/tomcat7 start
```

Installing Apache

Now copy or create the test files into `/home/solr/exampledocs` on your server using FileZilla, or Midnight Commander etc. I used the convention solr.pdf, solr.doc, solr.txt etc. for this how-to. If you point your browser to *http://yourserveripaddress* you should see a directory listing similar to (Figure 3). Perform a quick check to make sure you can download / open the example files. Also, check that collection4 has come up.

Step 2. Manually Test Tika and Create the Schema

We now want to ensure that Tika can extract the meta-data from the files. You should see the output similar to (Figure 4).

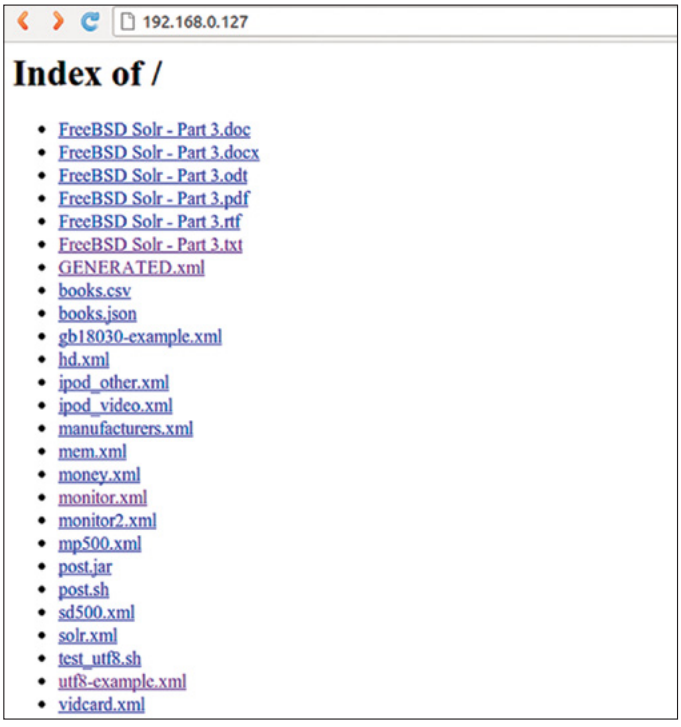


Figure 3. *Example docs file listing served via Apache*

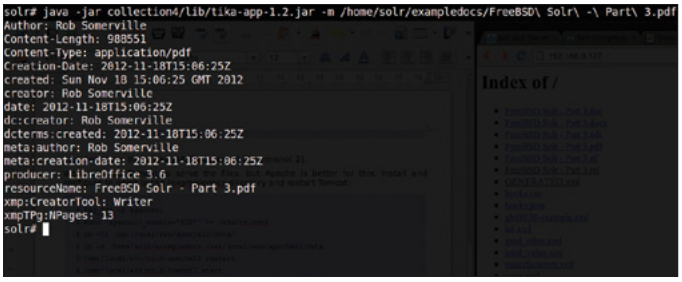


Figure 4. *Tika Metadata example*

```
# java -jar /tmp/solr/tika-app-1.2.jar -m \
    /home/solr/exampledocs/solr.pdf
```

Viewing metadata

Repeat with the different filetypes to ishow how Tika automatically extracts the different types of meta-data from the files. To see how Tika extracts the content from the PDF run the following command:

```
# java -jar /tmp/solr/tika-app-1.2.jar -t \
    /home/solr/exampledocs/solr.pdf
```

Viewing content

Stop Tomcat and edit `solrconfig.xml` to reflect (Listing 5)

```
# /usr/local/etc/rc.d/tomcat7 stop
# vi collection4/conf/solrconfig.xml
```

Editing solrconfig.xml (Listing 5).

Add the following field to schema.xml under fields (Listing 6).

Step 3. Restart tomcat and Load the Files

```
# /usr/local/etc/rc.d/tomcat7 stop
```

Restarting Tomcat

Using curl we will perform Stages 1 and 2 from (Figure 1) extracting both the content and the metadata from the files:

Listing 5. *XML*

```
<requestHandler name="/update/extract"
class="solr.extraction.ExtractingRequestHandler" >
  <lst name="defaults">
    <str name="fmap.a">links</str>
    <str name="fmap.div">ignored_</str>
    <str name="fmap.content">text</str>
    <str name="lowernames">true</str>
    <str name="uprefix">attr_</str>
    <str name="captureAttr">true</str>
  </lst>
</requestHandler>
```

Listing 6. *XML*

```
<dynamicField name="attr_*"
   type="text_general"
   indexed="true"
   stored="true"
   multiValued="true"/>
```

Figure 5. *Solr results with metadata*

```
# curl "http://localhost:8080/solr/collection4/update/
extract? \   literal.id=1&commit=true" -F "myfile=@/home/
                    solr/exampledocs/solr.pdf"
```

Extracting and parsing

You should see the following:

```
# <?xml version="1.0" encoding="UTF-8"?><response>
<lst name="responseHeader"><int   name="status">0
</int><int name="QTime">750</int></lst></response>
```

Solr response

Repeat for each document you want Solr to process, incrementing `literal.id` by one each time and replacing `solr.pdf` with `solr.rtf` and `solr.doc` etc.

Next Steps

To make this production ready, you need to accurately define the handlers used for each core and the schema for the files we want to search. A shell script could easily be used to load files *en masse*, it would just be a matter of writing some straightforward glue code in Perl or PHP as a basic front-end to pass the search query to Solr and return the metadata and a sample of the body content, along with the HTML link to the file.

Conclusion

In the past four articles we have demonstrated how powerful and flexible Solr, Tika and Nutch can be when used either side of the corporate firewall. When coupled with a well designed schema, it is a robust and scalable solution to the ever expanding challenge of integrating systems. While designed primarily as a search framework, applications from corporate website search through to document management are possible. With a little imagination (and if necessary some additional coding), Solr can truly be the central search facility for any organisation.

Table 2. *Required JAR files*

File-name	Components
tika-core-1.1.jar	Tika core library
tika-parsers-1.1.jar	Tika parsers
tika-app-1.2.jar	Tika application

Table 3. *Further reading*

Description	URL
Apache Tika website	*http://tika.apache.org/*
Tika download page	*http://tika.apache.org/download.html*

ROB SOMERVILLE

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

PostgreSQL: Schemas

This article provides an introduction to schemas, a feature of PostgreSQL that allow Database Administrators (DBAs) to organize their database objects, mainly tables, into name spaces in order to either avoid naming conflicts and better structure the database itself. All the examples shown here have been tested on a PostgreSQL 9.1 cluster running on a FreeBSD 8.2-RELEASE machine; all the example source code is available in a GitHub repository.

What you will learn…

- What schemas are and how to take advantage of them
- How to organize your database objects into schemas

What you should know…

- Basic shell commands
- Basic PostgreSQL concepts
- Database object creation (creating tables, indexes, and so on)

This article provides an introduction to *schemas*, a feature of PostgreSQL that allow *Database Administrators* (DBAs) to organize their database objects, mainly tables, into name spaces in order to either avoid naming conflicts and better structure the database itself. All the examples shown here have been tested on a PostgreSQL 9.1 cluster running on a FreeBSD 8.2-RELEASE machine; all the example source code is available in a GitHub repository.

What is a Schema?

As detailed in previous articles in this series, a PostgreSQL instance can manage a cluster of databases, all logically separated from each other. Within each database, the objects (tables, indexes, stored procedures, views, and so on) can be further split into named sets called *schemas*. Generally speaking, a schema can be thought as a "namespace" to contain database objects (mainly tables). Advantages of using schemas are mainly the followings:

- provide a clear structure of the database, keeping objects that are not strictly related to each other separated. As an example, a configuration table should not be in the same place as an accounting table, since the two tables serve different purposes;
- provides a level of granularity allowing a DBA to separately backup and restore a whole set of objects within a schema without having to touch the others, and to quickly set up permission grants on each set of objects addressing the whole schema they belong to;
- avoid naming conflicts, allowing objects within different schemas to have the same name.

In PostgreSQL each database object has to belong to a schema, therefore if the DBA or the developer does not explicitly specify any schema, PostgreSQL adopts a default schema named public. A database object is therefore fully qualified by its simple name and the name of the schema, with the schema coming first and with a dot '.' as separation, as follows:

```
schemaName.objectSimpleName
```

In a few cases the schema name can be omitted, and this is usually the case for the *public* schema (more on this later). In order to see the *public* schema in action consider the simple definition of the *magazine* table used in the previous articles' examples and shown in Listing 1. From a `psql(1)` terminal it is possible to see the definition of the table with the instrospection command `\d magazine`, as shown in Listing 2: please note that the system reports the table with the prefix *public*, and therefore the table is named *public.magazine* and not simply *magazine* as in the creation instruction of Listing 1.

Listing 1. *A magazine table definition*

```
DROP TABLE magazine CASCADE;
CREATE TABLE IF NOT EXISTS magazine(pk serial NOT NULL,
id text,
month int,
issuedon date,
title text,
PRIMARY KEY(pk),
UNIQUE (id)
);

TRUNCATE TABLE magazine;
INSERT INTO magazine (pk, id, month, issuedon, title)
VALUES(1,'2012-01', 1,       '2012-01-01'::date ,     'FreeBSD: Get Up To Date');

INSERT INTO magazine (pk, id, month, issuedon, title)
VALUES(2,'2011-12', 12,       '2012-04-01'::date , 'Rolling Your Own Kernel');


INSERT INTO magazine (pk, id, month, issuedon, title)
VALUES(3,'2011-11', 11,       '2011-01-01'::date, 'Speed Daemons');
```

Listing 2. *The full name of the table*

```
bsdmagdb=# \d magazine
                        Table "public.magazine"
  Column  |  Type   |                       Modifiers
----------+---------+-------------------------------------------------------
 pk       | integer | not null default nextval('magazine_pk_seq'::regclass)
 id       | text    |
 month    | integer |
 issuedon | date    |
 title    | text    |
Indexes:
    "magazine_pkey" PRIMARY KEY, btree (pk)
    "magazine_id_key" UNIQUE CONSTRAINT, btree (id)

bsdmagdb=# SELECT * FROM magazine;
 pk |   id    | month |  issuedon  |          title
----+---------+-------+------------+-------------------------
  1 | 2012-01 |     1 | 2012-01-01 | FreeBSD: Get Up To Date
  2 | 2011-12 |    12 | 2012-04-01 | Rolling Your Own Kernel
  3 | 2011-11 |    11 | 2011-01-01 | Speed Daemons
bsdmagdb=# SELECT * FROM public.magazine;
 pk |   id    | month |  issuedon  |          title
----+---------+-------+------------+-------------------------
  1 | 2012-01 |     1 | 2012-01-01 | FreeBSD: Get Up To Date
  2 | 2011-12 |    12 | 2012-04-01 | Rolling Your Own Kernel
  3 | 2011-11 |    11 | 2011-01-01 | Speed Daemons
```

Listing 3. *Creating the three magazine tables in separated schemas*

```
CREATE SCHEMA bsdmag;
CREATE TABLE IF NOT EXISTS bsdmag.magazine( /* as in
                    Listing 1 */);
CREATE SCHEMA pentestmag;
CREATE TABLE IF NOT EXISTS pentestmag.magazine( /* as in
                    Listing 1 */);
CREATE SCHEMA linuxmag;
CREATE TABLE IF NOT EXISTS linuxmag.magazine( /* as in
                    Listing 1 */)
```

Listing 4. *Inspecting available tables*

```
bsdmagdb=# \dn
   List of schemas
    Name    | Owner
------------+--------
 bsdmag     | bsdmag
 linuxmag   | bsdmag
 pentestmag | bsdmag
 public     | pgsql
bsdmagdb=# \d
              List of relations
 Schema |      Name       |   Type   | Owner
--------+-----------------+----------+--------
 bsdmag | magazine        | table    | bsdmag
 bsdmag | magazine_pk_seq | sequence | bsdmag
 public | articles        | table    | bsdmag
 public | articles_pk_seq | sequence | bsdmag
 public | listini         | table    | bsdmag
 public | listini_pk_seq  | sequence | bsdmag
 public | readers         | table    | bsdmag
 public | readers_pk_seq  | sequence | bsdmag
 public | test            | table    | bsdmag
 public | test_pk_seq     | sequence | bsdmag
```

Listing 5. *Ensuring all the tables are in the proper schema*

```
bsdmagdb=# \d public.magazine
                            Table "public.magazine"
  Column  |  Type   |                          Modifiers
----------+---------+-------------------------------------------------------------
 pk       | integer | not null default nextval('public.magazine_pk_seq'::regclass)
 id       | text    |
 month    | integer |
 issuedon | date    |
 title    | text    |
 . . .
bsdmagdb=# \d pentestmag.magazine
                            Table "pentestmag.magazine"
  Column  |  Type   |                            Modifiers
----------+---------+-------------------------------------------------------------------
 pk       | integer | not null default nextval('pentestmag.magazine_pk_seq'::regclass)
 id       | text    |
 month    | integer |
 issuedon | date    |
 title    | text    |
 . . .
bsdmagdb=# \d linuxmag.magazine
                            Table "linuxmag.magazine"
  Column  |  Type   |                          Modifiers
----------+---------+-------------------------------------------------------------
 pk       | integer | not null default nextval('linuxmag.magazine_pk_seq'::regclass)
 id       | text    |
 month    | integer |
 issuedon | date    |
 title    | text    |
```

Every SQL command understand the usage of a schema qualifier, and as shown in the bottom half of Listing 2, accessing the *magazine* table is possible either via the simple name *magazine* or the fully qualified name *public.magazine*.

Using Schemas

To better understand the advantages of using schemas, an ad-hoc example will be shown. Consider a database that will contain information about different published magazines, such as BSD Magazine, Linux Magazine, PenTest Magazine, and so on. All the magazines have a set of shared data, for instance the list of readers and authors, and a set of private data, such as each magazine title, issue, and so on. Instead of creating a different database for each set of data related to each magazine, the database will be only one, but each magazine will store its private data into a *magazine*-like table. This isn't the best real-world design, but it does suffice to explain how to solve some problems using schemas. Imagine also that there is a constraint to use the same table name for each magazine's private data, that is each magazine will store its data in a *magazine* called table within the same database. This means that in the database there will be at least three *magazine* tables, one for the BSD Magazine, one for the Linux Magazine, and one for the PenTest Magazine. The problems that arise from such a situation are (i) nameclashing, (ii) different permission handling, (iii) different backup strategies, while advantages are (i) the database is self contained, (ii) the structure of each set of data is exactly the same and (iii) each set can be profiled in a different way.

Using Schemas to Solve Name Conflicts

The first problem, name clashing, can be easily solved using schemas: it suffices to create a single schema for each magazine and to store the *magazine* table into this schema. A schema can be created with the `CREATE SCHEMA` command, which requires a name for the schema (that of course has to be unique within the database).

Listing 6. *Viewing the current search_path and modifying it*

```
bsdmagdb=# SHOW search_path;
  search_path
----------------
 "$user",public
bsdmagdb=# SET search_path TO linuxmag,public;
bsdmagdb=# SHOW search_path;
   search_path
------------------
 linuxmag, public
bsdmagdb=# \d
              List of relations
  Schema  |      Name       |   Type   | Owner
----------+-----------------+----------+--------
 linuxmag | magazine        | table    | bsdmag
 linuxmag | magazine_pk_seq | sequence | bsdmag
 public   | articles        | table    | bsdmag
 . . .
bsdmagdb=# SET search_path TO "$user",linuxmag,pentestma
                  g,bsdmag,public;
bsdmagdb=# SHOW search_path;
                 search_path
-----------------------------------------------
 "$user", linuxmag, pentestmag, bsdmag, public
(1 row)

bsdmagdb=# \d
             List of relations
 Schema |      Name       |   Type   | Owner
--------+-----------------+----------+--------
 bsdmag | magazine        | table    | bsdmag
 bsdmag | magazine_pk_seq | sequence | bsdmag
 public | articles        | table    | bsdmag
 . . .
```

Listing 7. *Inspecting which magazine table is available at any time depending on the search schema*

```
SELECT n.nspname, -- schema name
       c.relname, -- relation name
       c.oid,     -- relation oid
       pg_catalog.pg_table_is_visible( c.oid ) -- is the
                    schema in the search path?
FROM pg_class c
LEFT JOIN pg_catalog.pg_namespace n
     ON n.oid = c.relnamespace
WHERE c.relname = 'magazine'
ORDER BY n.nspname;
  nspname   | relname  |  oid   | pg_table_is_visible
------------+----------+--------+---------------------
 bsdmag     | magazine | 129115 | t
 linuxmag   | magazine | 129089 | f
 pentestmag | magazine | 129075 | f
 public     | magazine | 129047 | f
```

Listing 3 shows how to create each schema and how to insert the *magazine* table within each schema: please note that in the `CREATE TABLE` command the fully qualified name of the table is specified. Doing introspection on the database (see Listing 4) does not provide the result readers would expect: only the *bsdmag.magazine* table is shown in the list; there is no mention of *linuxmag.magazine* and *pentestmag.magazine,* and the *public.magazine* table seems to have disappeared. The truth is that all the above tables are in place, as shown in Listing 5, but the system does not report them as "directly available"; to understand why, another concept related to schemas has to be introduced: the *search path*.

The search path is a special PostgreSQL tunable that can be configured for each database user (and for each database session); its purpose is to instrument the searching within a set of schema locations for a non-qualified object name. Its usage is really similar to that of the *PATH* shell variable when searching for executables. By default, the schema search path is set to the tuple *$user* and *public*, which means that all the non-fully qualified objects will be searched first in a schema whose name is the same name of the database username that is currently running the session and then the *public* schema. As readers can see, this is a very elegant trick to allow each user to define a private table (or other kind of object) without having its name clash with instances of other users and without having to fully qualify it on each command (of course the object creation requires the fully qualified name). In all the examples shown in this article,

Listing 8. *Populating the three magazine tables*

```
INSERT INTO bsdmag.magazine (pk, id, month, issuedon, title)
VALUES(1,'2012-01', 1,       '2012-01-01'::date ,     'FreeBSD: Get Up To Date');
INSERT INTO bsdmag.magazine (pk, id, month, issuedon, title)
VALUES(2,'2011-12', 12,       '2012-04-01'::date , 'Rolling Your Own Kernel');
INSERT INTO bsdmag.magazine (pk, id, month, issuedon, title)
VALUES(3,'2011-11', 11,       '2011-01-01'::date, 'Speed Daemons');
INSERT INTO linuxmag.magazine (pk, id, month, issuedon, title)
VALUES(1,'2012-01', 1,       '2012-01-01'::date ,     'Understanding the Linux Kernel');
INSERT INTO linuxmag.magazine (pk, id, month, issuedon, title)
VALUES(2,'2011-12', 12,       '2012-04-01'::date , 'Gnome and Linux');
INSERT INTO linuxmag.magazine (pk, id, month, issuedon, title)
VALUES(3,'2011-11', 11,       '2011-01-11'::date, 'Interview with A. Seigo');
INSERT INTO linuxmag.magazine (pk, id, month, issuedon, title)
VALUES(4,'2011-10', 10,       '2011-01-10'::date, 'Compiling a kernel');
INSERT INTO linuxmag.magazine (pk, id, month, issuedon, title)
VALUES(5,'2011-09', 9,       '2011-01-09'::date, 'GNU Emacs');

SELECT count( m.pk) AS bsdmag_issues
FROM bsdmag.magazine m;
 bsdmag_issues
---------------
             3
SELECT count( m.pk) AS linuxmag_issues
FROM linuxmag.magazine m;
 linuxmag_issues
-----------------
               5
SELECT count( m.pk) AS pentestmag_issues
FROM pentestmag.magazine m;
 pentestmag_issues
-------------------
                 1
```

the database connection is made via the user *bsdmag*, and therefore the `$user` special tag is substituted by the string *bsdmag.* The *search_path* can be inspected with the `SHOW` command, and can be set with the `SET` command as shown in Listing 6. As readers can see, setting the *search_path* to the schemas *linuxmag* and *public* changes the behaviour of the instrospection on the database. Listing 6 now shows the *linuxmag.magazine* table as directly available.

It is worth noting that even setting the *search_path* to the full list of schemas that include `$user`, *linuxmag*, *pentestmag* and *public* does make more than one *magazine* table appear (while the system has one per each schema in the *search_path*). This is a design choice of PostgreSQL: in the case of name clashing the `\d` command shows only the first table that will be available in the *search_path.* This means that the other *magazine* tables are going to be hidden to the user until she qualifies them, and therefore the \d command reflects this state not showing a table that is not accessible without being fully qualified, even if the schema is in the *search_path*. PostgreSQL internally decides which tables are available in the case of name clashing using the *pg_table_is_visible* internal function, and users can inspect which *magazine* table is currently visible using `\d` with the query shown in Listing 7. Using the analogy of the *PATH* shell variable, the behaviour of the `\d` command is really similar to that of launching an executable using the *PATH* variable: the first executable hides the other in other directories.

The above discussion emphasizes how the order of entries in the *search_path* is important, since it dictates which schemas have to be searched first and, in case of naming conflicts, which objects are being targeted and which are not.

Finally, please take into account that the *search_path* can be configured for the whole cluster as explained in the Box 1.

In order to complete the discussion about the same table within different schemas, Listing 8 shows a simple population of each different table and the count of each issues for the three schemas; please note that all the tables exist independently from one another (Please note that, if all the tables the same structure, it could be worth using PostgreSQL inheritance (see previous articles on partitioning), but in order to keep the example simple and compact, the whole *magazine* table has been replicated within each schema).

Using Schemas to Apply Privileges

In order to demonstrate how schemas allows for a faster and easier set up of privileges, consider the case where two additional users have access to the database: *linuxmag_user* and *chief_editor* (see Box 2 on how to create users). The former is a user strictly related to the *linuxmag* set of database objects, while the latter is a user that

Listing 9. *Applying grant options using the schema facilities*

```
bsdmagdb=# REVOKE ALL PRIVILEGES ON ALL TABLES IN SCHEMA bsdmag FROM linuxmag_user;
bsdmagdb=# REVOKE ALL PRIVILEGES ON ALL TABLES IN SCHEMA pentestmag FROM linuxmag_user;
bsdmagdb=# REVOKE ALL PRIVILEGES ON ALL TABLES IN SCHEMA bsdmag FROM chief_editor;
bsdmagdb=# REVOKE ALL PRIVILEGES ON ALL TABLES IN SCHEMA pentestmag FROM chief_editor;
bsdmagdb=# REVOKE ALL PRIVILEGES ON ALL TABLES IN SCHEMA linuxmag FROM chief_editor;
bsdmagdb=# GRANT SELECT ON ALL TABLES IN SCHEMA bsdmag TO chief_editor;
bsdmagdb=# GRANT SELECT ON ALL TABLES IN SCHEMA pentestmag TO chief_editor;
bsdmagdb=# GRANT SELECT ON ALL TABLES IN SCHEMA linuxmag TO chief_editor;
bsdmagdb=# GRANT USAGE ON SCHEMA bsdmag TO chief_editor;
bsdmagdb=# GRANT USAGE ON SCHEMA pentestmag TO chief_editor;
bsdmagdb=# GRANT USAGE ON SCHEMA linuxmag TO chief_editor;

// when connected as linuxmag_user
bsdmagdb=> SELECT * FROM bsdmag.magazine;
ERROR:  permission denied for schema bsdmag
LINE 1: SELECT * FROM bsdmag.magazine;

// when connected as chief_editor
bsdmagdb=> SELECT count( b.pk ) FROM bsdmag.magazine b;
 count
-------
     3
```

should have access to all the sets for reading purposes (being *chief_editor,* not a database administrator). The security policy therefore is as follows:

- remove all privileges to all objects in the other schemas (bsdmag and pentestmag) from the linuxmag_user.
- provide privileges to chief_editor for all the schemas and all objects within the schema.

While such policy can be implemented with a set of privileges on each object's fully qualified name within a schema, the `GRANT` and `REVOKE` command in PostgreSQL are schema aware and allow the DBA to quickly target all objects in a schema. Therefore, as shown in Listing 9, it is possible to specify an *ALL TABLES IN SCHEMA* target to have all the grants applied recursively. Please note that the sequence of commands in Listing 9 is not the only way of achieving the security policy described above, and that it is necessary to provide the *usage* privilege on the schemas in order to allow the *chief_editor* user to "walk" the schema (something similar to the file permission schema with the directory executable bit).

The above example shows how to quickly apply custom privileges to a schema and a set of objects (tables) within it;

Listing 10. *An example backup of two of three schemas*

```
> pg_dump -n bsdmag -n linuxmag -U bsdmag bsdmagdb
--
-- PostgreSQL database dump
--

 . . .
--
-- Name: bsdmag; Type: SCHEMA; Schema: -; Owner: bsdmag
--
CREATE SCHEMA bsdmag;
ALTER SCHEMA bsdmag OWNER TO bsdmag;

--
-- Name: linuxmag; Type: SCHEMA; Schema: -; Owner: bsdmag
--

CREATE SCHEMA linuxmag;
ALTER SCHEMA linuxmag OWNER TO bsdmag;
 . . .
SET search_path = bsdmag, pg_catalog;
 . . .
```

since the schemas used in this article are made by a table and a sequence, there is not a huge advantage in using this `GRANT/REVOKE` syntax instead of addressing each fully qualified object. Nevertheless, in more complex deployment, it is quite common to have hundreds of tables within a single schema, and therefore having the capability of addressing a whole schema at once is a great time saver.

Using a Schema to Manage Custom Backup/Restore Policies

In the previous articles of this series readers have seen how `pg_dump(1)` and `pg_restore(1)` can be used to make a cold backup (that is a consistent backup at a specific time). Since the above tools are schema aware, it is possible to use the commands to backup/restore a specific schema among those in the database. The *-n* option of `pg_dump(1)` and `pg_restore(1)` can be used to specify one schema to backup; using multiple options allow the administrator to select multiple schemas at once, as shown in Listing 19 where only the *bsdmag* and *linuxmag* are going to be dumped. Of course it is important that the user that executes the dump have the privileges to access all objects in the selected schemas. It is worth noting how the dump performs the initial setup of the schemas and the search path for accessing objects during the restore phase.

Using Schema for Customization of Configuration

As described above, changing the *search_path* for a user allows the overriding of some database objects that have the same name. This allows for a per-user configuration and customization, since each user could be "pushed" to search for a particular object into a specific schema. To better understand, consider the Listing 11, that defines three versions of the same stored procedure *download_url* that, given the primary key of a *magazine* tuple returns a download URL for an issue. Each version of the function goes into one of the three schemas and returns a prefix that changes depending on the magazine it belongs to (Again, this is not the ideal design, but is used only to explain the schema facility). As shown in Listing 12, having different *search_path* allows a user to "see" different results, and therefore this can be used as a trick for differentiating users' profiles.

Suppose the *linuxmag_user* user has to be fully customized so that when calling the `download_url()` stored procedure the `linuxmag.download_url()` is effectively called; there are two ways of achieving this:

- setting the search_path of the user so that the linuxmag schema is the first entry;
- configure a linuxmag schema with a stored procedure that wraps `linuxmag.download _ url()`.

Listing 11. *Three procedure placed each in a different schema*

```
CREATE OR REPLACE FUNCTION bsdmag.download_url(
                    magazine_pk integer )
RETURNS text
AS
$BODY$
DECLARE
        magazine_id     text;
BEGIN

        -- get the magazine id
        SELECT id
        INTO   magazine_id
        FROM   magazine
        WHERE  pk = magazine_pk;

        IF magazine_id IS NULL THEN
           RETURN '';
        END IF;
    RAISE LOG 'bsdmag.download_url()';
        -- this is the part that changes depending on
                    the schema
        RETURN 'http://bsdmag.org/download/' ||
                    magazine_id || '.pdf';


END;
$BODY$
LANGUAGE plpgsql;



CREATE OR REPLACE FUNCTION linuxmag.download_url(
                    magazine_pk integer )
RETURNS text
AS
$BODY$
DECLARE
        magazine_id     text;
BEGIN

        -- get the magazine id
        SELECT id
        INTO   magazine_id
        FROM   magazine
        WHERE  pk = magazine_pk;

        IF magazine_id IS NULL THEN
           RETURN '';
        END IF;

    RAISE LOG 'linuxmag.download_url()';
        -- this is the part that changes depending on
                    the schema
        RETURN 'http://linuxmag.org/download/' ||
                    magazine_id || '.pdf';


END;
$BODY$
LANGUAGE plpgsql;



CREATE OR REPLACE FUNCTION pentestmag.download_url(
                    magazine_pk integer )
RETURNS text
AS
$BODY$
DECLARE
        magazine_id     text;
BEGIN

        -- get the magazine id
        SELECT id
        INTO   magazine_id
        FROM   magazine
        WHERE  pk = magazine_pk;

        IF magazine_id IS NULL THEN
           RETURN '';
        END IF;
    RAISE LOG 'pentestmag.download_url()';
        -- this is the part that changes depending on
                    the schema
        RETURN 'http://pentestmag.org/download/' ||
                    magazine_id || '.pdf';


END;
$BODY$
LANGUAGE plpgsql;
```

The first way is left as an exercise for the reader, while the latter is shown in Listing 13: first a schema with the same name of the user has to be created, and then a wrapper function is placed in such schema. It is even possible to move the stored procedure from its original schema to the user's, but this could make it impossible for other users to access the same procedure (in this case, security restrictions on the per-user schema are enforced). Finally, an alias to the right *magazine* table has to be set up, and this is done via a view.

As shown in Listing 14, calling the `download_url()` procedure as the *linuxmag_user* now redirects to the `linuxmag.download_url()` procedure in the view that in turn uses the *linuxmag.magazine* table. As shown, the per-user schema configuration is a very powerful feature that requires a little extra effort for a correct set up, but can turn out to be a fundamental capability to allow portability. It is clear that all the above function definitions and per-schema object set up can be automated using scripting and stored procedures (see previous articles), reducing the DBA load.

Operating on a Whole Schema

A DBA can perform other interesting operations on a schema and all its contained objects at once, the most

Listing 12. *Changing the schema search_path allows a user to get different behaviours*

```
bsdmagdb=# SELECT download_url( 1 );
LOG:  bsdmag.download_url()
              download_url
----------------------------------------
 http://bsdmag.org/download/2012-01.pdf

bsdmagdb=# SET search_path TO pentestmag,linuxmag,bsdmag
                   ,public;
bsdmagdb=# SELECT download_url( 1 );
LOG:  pentestmag.download_url()
               download_url
--------------------------------------------
 http://pentestmag.org/download/2012-07.pdf
```

Listing 13. *Setting up objects for a complete per-user customization*

```
CREATE SCHEMA linuxmag_user;

-- remove all privileges to all other users
REVOKE ALL PRIVILEGES ON SCHEMA linuxmag_user FROM
                   PUBLIC;

-- grant all privileges to the running user
GRANT ALL PRIVILEGES ON SCHEMA linuxmag_user TO
                   linuxmag_user;

-- grant usage for the schema target of the functions
GRANT USAGE ON SCHEMA linuxmag TO linuxmag_user;
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA linuxmag TO
                   linuxmag_user;

-- create a wrapper function
CREATE OR REPLACE FUNCTION linuxmag_user.download_url(
                   magazine_pk integer )
RETURNS text
AS
$BODY$
DECLARE
BEGIN
        RAISE LOG 'linuxmag_user.download_url()';
        RETURN linuxmag.download_url( magazine_pk );


END;
$BODY$
LANGUAGE plpgsql;

-- create a wrapper view for the magazine table
CREATE OR REPLACE VIEW linuxmag_user.magazine
AS
SELECT *
FROM linuxmag.magazine;
```

Listing 14. *Example of calling the download_url() procedure as linuxmag_user user*

```
bsdmagdb=> SELECT current_user;
 current_user
---------------
 linuxmag_user

bsdmagdb=> SELECT download_url( 1 );
LOG:  linuxmag_user.download_url()
LOG:  linuxmag.download_url()
CONTEXT:  PL/pgSQL function "download_url" line 5 at RET
                   URN
LOG:  duration: 5.670 ms  statement: SELECT download_
                   url( 1 );
               download_url
----------------------------------------
 http://linuxmag.org/download/2012-01.pdf
```

common being dropping the whole schema and renaming the schema. The `DROP SCHEMA` command is used to drop a schema; since the schema can contain different objects the database prevents accidental schema deletion by informing the user about dependencies, as shown in the top half of Listing 15. In order to recursively drop a schema and all its content it is required to use the command `DROP SCHEMA … CASCADE`, as shown in the bottom half of Listing 15. The system will inform the user about the objects that are going to be dropped due to the cascade option.

To rename a schema the special command `ALTER SCHEMA...RENAME` can be used, as shown in Listing 16. The

Listing 15. *Dropping the linuxmag schema*

```
bsdmagdb=# DROP SCHEMA linuxmag;
ERROR:  cannot drop schema linuxmag because other objects depend on it
DETAIL:  table linuxmag.magazine depends on schema linuxmag
view linuxmag_user.magazine depends on table linuxmag.magazine
function linuxmag.download_url(integer) depends on schema linuxmag
HINT:  Use DROP ... CASCADE to drop the dependent objects too.
bsdmagdb=# DROP SCHEMA linuxmag CASCADE;
NOTICE:  drop cascades to 3 other objects
DETAIL:  drop cascades to table linuxmag.magazine
drop cascades to view linuxmag_user.magazine
drop cascades to function linuxmag.download_url(integer)
DROP SCHEMA
```

Listing 16. *Renaming a schema and changing the owner*

```
bsdmagdb=# ALTER SCHEMA pentestmag RENAME TO archived_pentestmag;
bsdmagdb=# ALTER SCHEMA archived_pentestmag OWNER TO linuxmag_user;
```

Listing 17. *Creation and inspection of a temporary table*

```
bsdmagdb=> CREATE TEMPORARY TABLE temp_table( pk integer NOT NULL PRIMARY KEY, title text );
bsdmagdb=> \d temp_table
 Table "pg_temp_1.temp_table"
 Column |  Type   | Modifiers
--------+---------+-----------
 pk     | integer | not null
 title  | text    |
Indexes:
    "temp_table_pkey" PRIMARY KEY, btree (pk)

bsdmagdb=> SELECT n.nspname, -- schema name
       c.relname, -- relation name
       c.oid,     -- relation oid
       pg_catalog.pg_table_is_visible( c.oid ) -- is the schema in the search path?
FROM pg_class c
LEFT JOIN pg_catalog.pg_namespace n
     ON n.oid = c.relnamespace
WHERE c.relname = 'temp_table'
ORDER BY n.nspname;
nspname  |  relname   |  oid   | pg_table_is_visible
----------+------------+--------+---------------------
 pg_temp_1 | temp_table | 129144 | t
```

`ALTER SCHEMA` command can also be used to change the schema owner. It is worth noting that changing the owner of a schema will not affect the ownership of contained objects; they will keep their previous owner. To do it massively a script that iterates on *pg_class* or *pg_tables* (in the case of only tables) and issues more *alter* commands has to be used.

Box 1. Setting the search_path for all clients

The PostgreSQL configuration file postgresql.conf contains a definition of the variable search_path that can be used to set the search_path for each client that connects to the cluster. Specify the string that defines the search path, as used in the SET command (with quotes), to make the clients receive the path as their initial search path:

```
search_path = '"$user",public'
```

It is interesting to note that this change affects the whole cluster, not a single database, and therefore for very specific settings it is better to work on the single client connection.

Box 2. Creating database users

Database users can be created using the CREATE ROLE command, launched by a database superuser. The command provides many options, therefore the following is just an example of how to quickly create a user:

```
bsdmagdb=# CREATE ROLE chief_editor WITH LOGIN PASSWORD 'chief';
```

Box 3. Moving objects across schemas

PostgreSQL allows a database object to be moved across different schemas, of course assuming the user that moves the objects have the rights to do the delete/insert operation in the source/target schemas. The schema migration is done using the `ALTER...SET SCHEMA` commands, available for any kind of object that can be enclosed into a schema (e.g., tables, stored procedures, etc.). As an example, to move the table public.my_table from the schema public to the schema my_schema it is possible to use the following command:

```
bsdmagdb=# ALTER TABLE public.my_table SET SCHEMA my_schema;
```

An exception to the above is for temporary objects, like temporary tables, that cannot be moved out from their schema:

```
bsdmagdb=# ALTER TABLE temp_table SET SCHEMA linuxmag;
```

ERROR: cannot move objects into or out of temporary schemas

On The Web

- PostgreSQL official Web Site: *http://www.postgresql.org*
- ITPUG official Web Site: *http://www.itpug.org*
- Oddity with \d and pg_table_is_visible: *http://archives.postgresql.org/pgsql-hackers/2007-09/msg00205.php*
- GitHub Repository containing the source code of the examples: *https://github.com/fluca1978/fluca1978-pg-utils*

Temporary Tables

Temporary tables are a feature that allows a database to contain a table that will not persist on disk at any time, and therefore will not be recoverable (it is not written at all in the WAL logs) and cannot be backed up. The idea is to define a table from scratch to store in it some volatile location for testing or to create a materialized set of data to speed up later computations. Temporary tables are not strictly related to schemas, but their implementation is based on schemas. When a user creates a temporary table using the `CREATE TEMPORARY TABLE` command (as shown in Listing 17), the table is placed in a special schema named after the progressive connection number *pg_temp_X* (being X the number of the connection to the cluster). As shown in Listing 15, the temporary table *temp_table* is qualified by the name *pg_temp_1.temp_table*. The fully qualified table is defined as any other table, and therefore is available to other users and sessions, at least until is destroyed. The *search_path* of the user that has defined the table is not changed, however the user is able to access the table even using the simple name. This is due to the `pg_table_is_visible()` internal function returning true for each temporary table defined by the user himself (see bottom half of Listing 17). The same is not true for all other users, that are required to access the table using the fully qualified name.

It is worth noting that temporary schemas *pg_temp_X* are sealed: objects cannot be moved into or out of them (see Box 3). However it is possible to create objects into a *pg_temp_X* schema, even if such kind of objects will be destroyed when the client disconnects.

Summary and Coming Next

This article introduced the concept of schema, a very powerful abstraction that allows DBAs to organize database objects into coherent and interrelated packages. In the next article the management of users, groups and permissions within PostgreSQL will be shown.

LUCA FERRARI

Luca Ferrari lives in Italy with his wife and son. He is an Adjunct Professor at Nipissing University, Canada, a co-founder and the vice-president of the Italian PostgreSQL Users' Group (ITPUG). He simply loves the Open Source culture and refuses to log-in to non-Unix systems. He can be reached on line at http://fluca1978.blogspot.com.

Great Specials

On FreeBSD & PC-BSD Merchandise

Give us a call & ask about our
SOFTWARE BUNDLES
1.925.240.6652

$39.95
FreeBSD 9.0 Jewel Case CD Set
or FreeBSD 9.0 DVD

$29.95
PC-BSD 9.0 DVD

$49.95 Save a BUNDLE!
The PC-BSD 9.0 Users Handbook
PC-BSD 9.0 DVD

$99.95
The FreeBSD CD **or** DVD Bundle

Inside each CD/DVD Bundle, you'll find:
FreeBSD Handbook, 3rd Edition
Users Guide FreeBSD Handbook, 3rd Edition, Admin Guide
FreeBSD 9.0 CD or DVD set
FreeBSD Toolkit DVD

Stylish Dress Attire
Look Your Professional Best

Comfy Hoodies
Stay Warm in Pullovers & Zip Ups

T-Shirts
Lots of Styles to Choose From

FreeBSD 9.0 Jewel Case CD/DVD $39.95

CD Set Contains:

- **Disc 1:** Installation Boot LiveCD (i386)
- **Disc 2:** Essential Packages Xorg, GNOME2 (i386)
- **Disc 3:** Installation Boot LiveCD (amd64)
- **Disc 4:** Essential Packages Xorg, GNOME2 (amd64)

FreeBSD 8.2 CD $39.95
FreeBSD 8.2 DVD $39.95

FreeBSD Subscriptions

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

FreeBSD Subscription, start with CD 9.0 ..... $29.95
FreeBSD Subscription, start with DVD 9.0 ..... $29.95
FreeBSD Subscription, start with CD 8.2 ..... $29.95
FreeBSD Subscription, start with DVD 8.2 ..... $29.95

PC-BSD 9.0 DVD (Isotope Edition)

PC-BSD 9.0 DVD $29.95
PC-BSD Subscription $19.95

The FreeBSD Handbook

The FreeBSD Handbook, Volume 1 (User Guide) $39.95
The FreeBSD Handbook, Volume 2 (Admin Guide) $39.95

The FreeBSD Handbook Specials

The FreeBSD Handbook, Volume 2 (Both Volumes) $59.95
The FreeBSD Handbook, Both Volumes & FreeBSD 9.0 ..... $79.95

PC-BSD 9.0 Users Handbook $24.95

BSD Magazine $11.99

The FreeBSD Toolkit DVD $39.95

FreeBSD Mousepad $10.00

FreeBSD & PCBSD Caps $20.00

BSD Daemon Horns $2.00

Bundle Specials!
Save $$$

Just Plain Fun
Mousepads & Novelty Horns

BSD Magazine
Available Monthly

For even **MORE** items
visit our website today!

www.FreeBSDMall.com

A Simple
DNS-DHCP Server
for Small Business Network with Dnsmasq

An alternative to BIND and ISC-DHCP for small corporate and home networks to simplify the management of names and IP addresses in the LAN and from the Internet.

What you will learn…
- In this paper we will learn to setup and manage a Small Business DNS/DHCP server.

What you should know…
- Basic BSD Networking Setup and basic Networking structure knowledge.

For connecting a PC to a LAN (Local Area Network) one needs some basic parameters, which identifies it uniquely within the network itself, namely:

IP address – subnet mask (netmask) – gateway address – a DNS server to resolve domain names (Domain Name System).

A DNS Server "translates" the domain name (for example pippo.com) to an IP addresses (such as 192.168.10.1), and it is the only one that allows you to uniquely identify the machines within the networks, including the Internet.

A DNS server configuration is usually only used in large networks, and almost never in small LANs, in which the resolution service domain names relies on an external server. In this article we will see how to set the automatic configuration of network parameters on each machine connected to the LAN, including the resolution of domain names to the hosts in it. What you will get will be a service able to ensure:

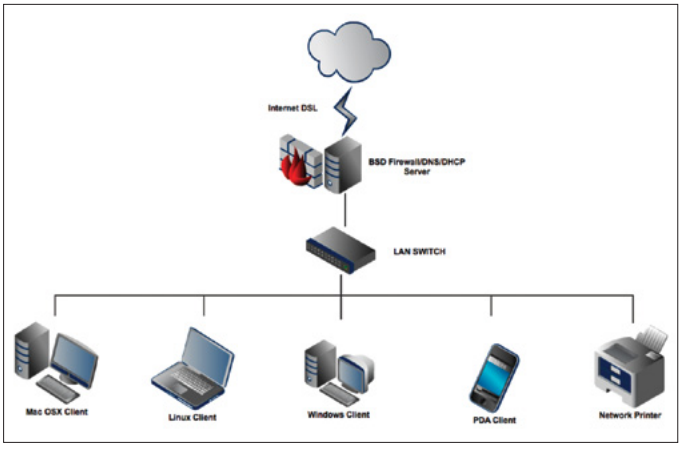


Figure 1. *A typical Small Business LAN Scenario*

- A DNS configuration of machines "behind" the firewall simple and independent by a DNS provider
- Timeout immediate for clients in the absence of the internet
- Names of local machines centralized on the firewall's file /etc/hosts automatically propagated.
- DHCP service switch with DHCP leases static and dynamic IP ranges and multiple.
- Caching internet addresses (A records and AAAA records and PTR records) with improved network performance.
- Support for MX and SRV records type and ability to provide the MX record for some or all machines on the local network, including the resolution of domain names to the hosts in it.

For the complete setup of a LAN DNS/DHCP service one usually needs BIND (Berkeley Internet Name Domain) and ISC-DHCPD (Dynamic Host Configuration Protocol), both available at *www.isc.org*. BIND is "the"

DNS server used by many ISPs and Domain Register, which, however, has the flaw that its configuration is not simple, is based mainly on the creation of cryptic text files for different areas (hostname, domain, or sections of domains) and has a complex base configuration, also for simple scenarios (Figure 2).

DHCP, instead, is composed of a client / server system and is responsible for the automatic configuration of the network parameters. In practice, DHCP clients (PCs connected to the network) send the request to the server to get configuration parameters.

On the other hand, the DHCP server receives the request and, based on the MAC address (hardware address

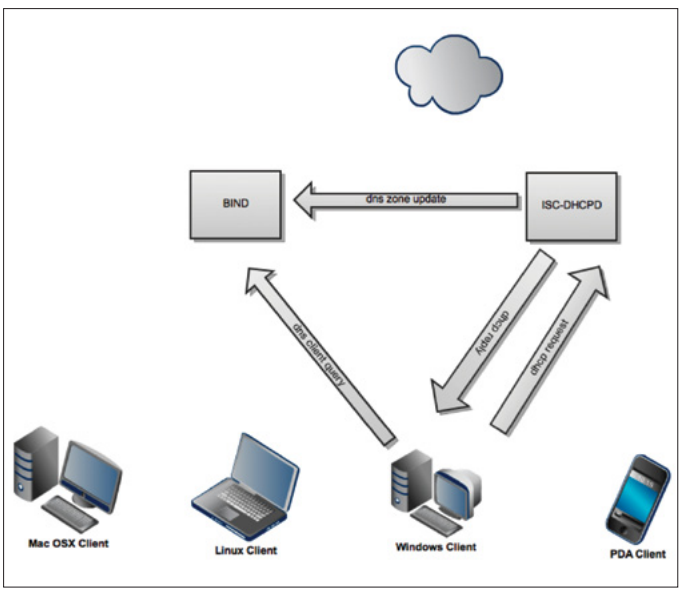


Figure 2. *Bind + ISC-DHCPD Operating Diagram*

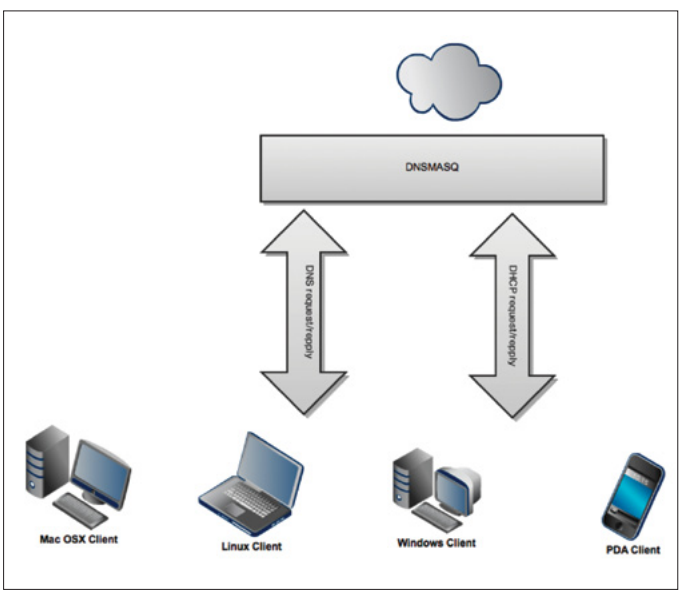


Figure 3. *Dnsmasq Operating Diagram*

that is unique to each network adapter) of the client, responds by providing the network parameters necessary to use the local network and the Internet.

The local IP address can also be uniquely associated with a MAC address. In this way, the user (the machine) will always have the same address instead of a randomly chosen among those available. In addition, the DHCP server can be configured to communicate to the BIND hostname and address of the machine that provided the connection parameters in order to automatically update the zone file on the local network.

Dnsmasq, the Alternative for Small Offices

Dnsmasq, the software that we will manage, is basically a mini DNS server that can resolve the names of computers on the local network and at the same time provide a DHCP service, although it is not suitable when you need advanced features such as DNS configuration master / slave or to manage a very large number of "zones" (Figure 3).

Pre-installation Steps

We must do some presets before starting the installation of the Dnsmasq server, assuming the same pc that hosts it will act as a gateway firewall for our LAN.

On FreeBSD

We must correctly setup `/etc/rc.conf` to have an operating network setup, in particular we need to configure the LAN network interface with a static IP, as shown in Llisting 1. Set the DNS in `/etc/resolv.conf`:

Listing 1. *A basic /etc/rc.conf file for a classic Dnsmasq Server FW*

```
### rc.conf
# WAN static connection
# ifconfig_xl0="inet 10.0.0.254 netmask 255.255.255.0"
#  WAN dhcp connection
ifconfig_xl0="dhcp"

# LAN connection
ifconfig_xl1="inet 172.16.0.1 netmask 255.255.255.0"

# Default gateway
# Set the gateway for static connection
# defaultrouter="10.0.0.1"

# Enable ip forward
gateway_enable="YES"

# Hostname
hostname="fw.bsdmag.lan"
### end rc.conf
```

```
nameserver   172.16.0.1
nameserver    10.0.0.1 # for static connection
```

On OpenBSD

We must correctly setup the system to have an operating network setup, in particular we need to configure the LAN network interface with a static IP, as shown below:

/etc/hostname.xl1

```
inet 172.16.0.1 255.255.255.0  172.16.0.255
  # LAN NETWORK SETUP
```

/etc/hostname.xl0

```
dhcp # WAN DHCP NETWORK SETUP
```

We must enable port forwarding by uncommenting this line in `/etc/sysctl.conf`:

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

Setup the OpenBSD box's default gateway editing `/etc/mygate`:

```
172.16.0.1
```

Setup the OpenBSD box's hostname editing `/etc/myname`:

```
fw.bsdmag.lan
```

Setup the `/etc/rc.conf.local` services

```
dhcpd_flags="xl0"
#pf=NO
```

Set the DNS in `/etc/resolv.conf`:

```
nameserver   172.16.0.1
nameserver   <YOUR_ISP_NAMESERVER>
```

Setup via Source Code or Using Packages

One may install Dnsmasq on any compatible Unix platform. Just choose whether to use the installation from source (the latest release is dnsmasq-2.63) or from a package.

Dnsmasq on OpenBSD 5.1 – 5.2

On OpenBSD 5.1 we need to do a little hack: First navigate to the rc.d directory and download the start script:

```
cd /etc/rc.d
curl -o dnsmasqd \ http://ftp.openbsd.org/ports/net/
                   dnsmasq/pkg/dnsmasq.rc.
```

Setup like this line 5:

```
daemon="/usr/local/sbin/dnsmasq"
```

Start the service using:

```
/etc/rc.d/dnsmasqd start
```

Listing 2. *The classic /etc/resolv.conf file of a Dnsmasq Server*

```
# OpenDNS DNS SERVERS
nameserver 127.0.0.1
nameserver 208.67.222.222
nameserver 208.67.220.220
# DNS SERVERS GOOGLE
nameserver 8.8.8.8
nameserver 8.8.4.4
```

Listing 3. *A simple dnsmasq.conf file*

```
no-dhcp-interface = xl0
addn-hosts = /etc/dnsmasq-hosts
no-hosts
local = /bsdmag.lan/
interface = xl1
expand-hosts
domain = bsdmag.lan
dhcp-range = 172.16.0.2,172.16.0.200,12h
dhcp-option = option: router, 172.16.0.1
dhcp-option = 44,172.16.0.1
dhcp-option = 45,172.16.0.1
dhcp-option = 46.8
dhcp-option = 47
dhcp-option = 6,172.16.0.1
mx-host = mail.bsdmag.lan, 50
mx-target = mail.bsdmag.lan
localmx
log-queries
log-dhcp
```

If it says it starts in `rc.conf.local` add:

```
pkg_scripts="dnsmasqd"
```

On OpenBSD 5.2 to get the software last release one may install from sources:

```
# cd / tmp
# wget-c http://www.thekelleys.org.uk/dnsmasq/
                    dnsmasq-2.63.tar.gz
# tar-xvzf dnsmasq-2.63.tar.gz
# cd dnsmasq-2.63
# make install
```

If one dislikes to install from sources, it's possible to use the packaged version, but it's not updated.

```
# pkg_add -r -v dnsmasq
```

Dnsmasq on FreeBSD

One can install Dnsmasq by using the ports collection:

```
$ cd /usr/ports/dns/dnsmasq/
$ su
# make install clean
```

or by using `pkg _ add` with the command

```
# pkg_add -r -v dnsmasq
```

The Dnsmasq script will be installed in `/usr/local/etc/rc.d`, and to get it to start at boot time, add this line to `/etc/rc.conf`:

```
dnsmasq_enable="YES"
```

Then start Dnsmasq:

```
$ su

# /usr/local/etc/rc.d/dnsmasq start
```

Basic LAN Name Configuration

Dnsmasq normally uses only the system file `/etc/hosts` on the PC running the service, associating the following names to IP addresses, but here we will use a static address file just by putting them in a new file `/etc/dnsmasq-hosts` (as specified by the parameter addn-hosts) in the form without a domain. We do not use the default `/etc/hosts` file in this scenario for preventing DNS server to resolve the "private" names that one may put here, for example "localhost".

```
# cat /etc/dnsmasq-hosts
…
172.16.10.1 proxyserver firewall
172.16.10.2 vpnserver
…
```

As you can see from this extract of the hosts file, a fully qualified domain name or more “short names” can be associated with each IP.

Basic configuration of external DNS server

To resolve names outside your local network, Dnsmasq uses the DNS servers in `/etc/resolv.conf`, which is structured as follows: Listing 2.

Configuring Services

Now that the external DNS servers is configured, it's time to setup the file `dnsmasq.conf`. This file in OpenBSD setup is stored in `/etc` folder, in FreeBSD setup is in `/usr/local/etc` folder.

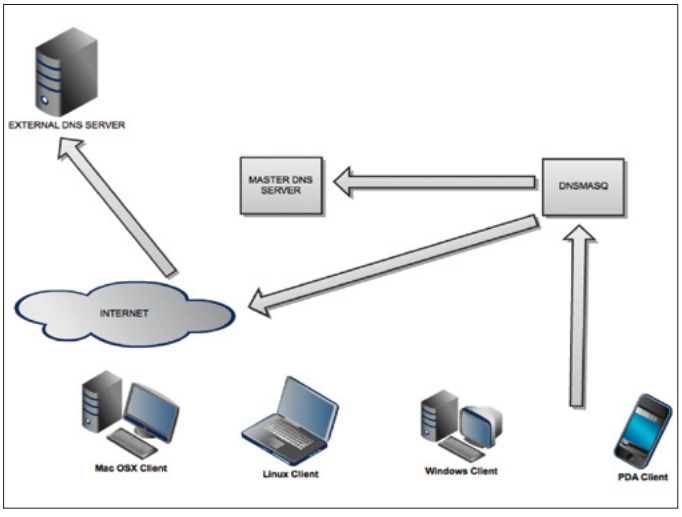


Figure 4. *Possible Dnsmasq Queries Scenario*

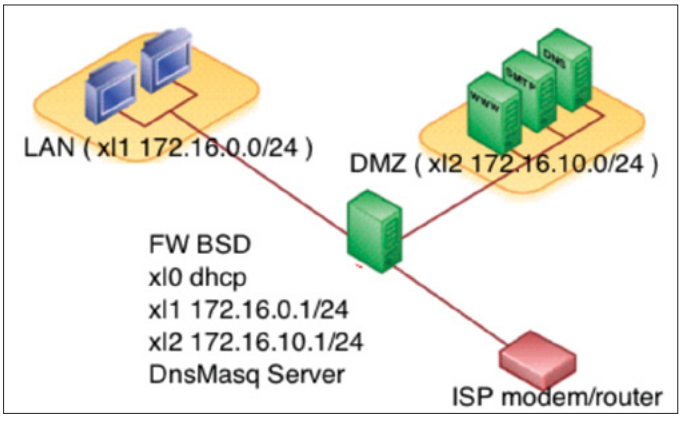


Figure 5. *A possible Dnsmasq LAN+DMZ Scenario*

The file is well commented. We may set the following parameters for a starting configuration, but We must remember to setup network interfaces and IP range consistent with our hardware and network architecture, especially lines as “interfaces”, “no-dhcp-interface” and “dhcp-range”: Listing 3.

And we will explain all the configuration sets:

- local = / .... / adds declarations for forward and reverse DNS queries
- expand-hosts add the domain to simple names
- domain = ... specifies DNS domains for the DHCP server
- no-dhcp-interface = ... Which interface set on the server must not listen to
- dhcp-range = ... sets up the DH CP ip pool with a default lease time
- dhcp-option = option: router, ... sets up the LAN gateway
- log-queries logs requests dns
- log-dhcp logs dhcp requests
- addn-hosts = /etc/dnsmasq-hosts file specifies the alternative hosts for the resolution of local machines
- no-hosts specify to not use the hosts file for name resolution
- dhcp-option = 44,172.16.0.1 Set NetBIOS-over-TCP/IP nameservers aka WINS servers
- dhcp-option = 45,172.16.0.1 netbios datagram distribution server
- dhcp-option = 46.8 netbios node type
- dhcp-option = 47 empty netbios scope.
- dhcp-option = 6,172.16.0.1 setup the lan dns server
- mx-host = mail.bsdmag.lan, 50 useful for directing mail from systems on a LAN to a central server
- mx-target = mail.bsdmag.lan specify the default target for the MX record returned by Dnsmasq
- localmx return an MX record pointing to the host Given by mx-target for each local machine

To see all the parameters "dhcp-options" is possible run the command:

```
# dnsmasq – help dhcp
```

For explanations on the individual parameters, refer to the online help

```
# man dnsmasq
```

As we have seen, to enable the DHCP server is necessary to define at least one of its essential directives such as, for example, "dhcp-range":

```
no-dhcp-interface=xl0
dhcp-range=172.16.0.2,172.16.0.200,12h
dhcp-option=option:router,172.16.0.1
```

The first line excludes a network interface from DHCP, which, however, will continue to be valid for DNS requests. The second line defines the range of addresses to be assigned dynamically.

Advanced Configuration

The default behavior is to communicate to clients the Dnsmasq server IP as a common gateway and DHCP/DNS services provider. However, the computer running Dnsmasq is not always a gateway, but we can force the correct gw adress with the directive

```
dhcp-option=option:router,172.16.0.1.
```

In a more complex scenario we may have two internal network interfaces (xl1 for LAN and xl2 for DMZ), and one external (xl0 connected to the Internet), to serve a small business with a more complex topology that publish service to the Internet (Figure 5).

For a complete list of options is necessary to refer to the online dnsmasq manual: *http://leaf.sourceforge.net/doc/man/dnsmasq.8.html* (Listing 4).

Let's now explain the rows of the configuration. If we want to associate fixed addresses to some machines, we record their MAC address and add one of the following directives:

```
dhcp-host = 00: c6: 77:26:26: server1, 172.16.10.40

except-interface= …
```

Listing 4. *A more complex configuration file for Dnsmasq*

```
no-dhcp-interface = xl0
except-interface=xl0
addn-hosts = /etc/dnsmasq-hosts
bogus-priv
resolv-file=/etc/resolv.conf.dnsmasq
no-hosts
local = /bsdmag.net/
expand-hosts
domain = bsdmag.net

# LAN DHCP DNS Service listening on interface xl1
interface = xl1
dhcp-range =xl1,172.16.0.2,172.16.0.200,12h
dhcp-option =xl1,option: router, 172.16.0.1
dhcp-option =xl1,44,172.16.0.1
dhcp-option =xl1,45,172.16.0.1
dhcp-option =xl1,46.8
dhcp-option =xl1,47
dhcp-option =xl1,6,172.16.0.1

# DMZ DHCP DNS Service listening on interface xl2
interface = xl2
dhcp-range =xl2,172.16.10.2,172.16.10.200,12h
dhcp-option =xl2,option: router, 172.16.10.1
dhcp-option =xl2,44,172.16.10.1
dhcp-option =xl2,45,172.16.10.1
dhcp-option =xl2,46.8
dhcp-option =xl2,47
dhcp-option =xl2,6,172.16.10.1


# Generic stuff
mx-host = mail.bsdmag.net, 50
mx-target = mail.bsdmag.net
localmx
log-queries
log-dhcp
cache-size=2048
log-facility=/var/log/dnsmasq/dnsmasq.log
dhcp-leasefile=/var/log/dnsmasq/dnsmasq.leases

# Static DHCP Host List
dhcp-host = 00: c6: 77:26:26: server1, 172.16.10.40
dhcp-host = 02: f6: 56:16:32: server2, 172.16.10.41
```

Listing 5. *A /etc/resolv.conf file for the advanced Dnsmasq setup*

```
search bsdmag.lan
nameserver 127.0.0.1
nameserver 8.8.8.8
nameserver 8.8.4.4
```

We do not listen on the specified interface.

bogus-priv
We do not propagate the addresses referred to areas that are not rotated and all the reverse lookup for the private subnet not present in `/etc/dnsmasq-hosts` or DHCP lease obtain "host not found" instead of being forwarded to the external dns.

resolv-file= …
we get all the IP addresses of the upstream nameservers from `<file>` instead of using `/ etc / resolv.conf`.

cache-size= ...
It's the size of Dnsmasq's cache. The default is 150 names. Setting the cache size to zero disables caching.

log-facility=...
set the facility to which Dnsmasq will send various logs, if the variable contains at least one '/' character is assumed to redirect the output to a file instead of syslog.

dhcp-leasefile= …
this is the file where Dnsmasq file keeps track of ip delivered to clients.

If you have multiple interfaces that offer DNS / DHCP must specify the prefix “ifname,” just before the network parameters in the Directive "dhcp-option", as in the previous file.

Now we modify our /etc/resolv.conf

```
# emacs /etc/resolv.conf
```

which must contain the DNS server queries, ie, itself and secondly the DNS provider which will make the cache (for example, 8.8.8.8) (Listing 5).

Logging Analysis

As shown there are four main parameters to configure logs:

```
log-queries
log-dhcp
log-facility=/var/log/dnsmasq/dnsmasq.log
dhcp-leasefile=/var/log/dnsmasq/dnsmasq.leases
```

But if we experience strange behavior we'll be able to analyze DHCP packets for monitoring or debugging purposes by using tcpdump and dhcpdump programs. The last one provides a tool for visualization of DHCP packets for analyzing DHCP server responses in tcpdump style. We may compare the output of tcpdump and dhcpdump commands, but here we will only explain the syntax, to let you see the differences.

Here we use tcpdump to capture DHCP output

```
# tcpdump -lenx -i xl1 -s 1500 port bootps or port bootpc
```

and here dhcpdump

```
#dhcpdump -i xl1
```

What do you think about the differences??

Conclusions

A real example of small LAN business network are the so called “SoHo” (single office/home office SOHO), namely a category of businesses that has 1 to 10 employees, but this is only the staring point. In fact, there are examples of deployable environment for Dnsmasq configurations used for more than 1000 hosts. On the other side of the coin there are still some limitations, such as a very basic support for IPv6 router advertisements for DHCPv6 to work and the inability to serve many zone files (many domains), but this project brought us many surprises in time and will only get better. Knowing the strengths and limits of this daemon, a network administrator can now decide whether to install Dnsmasq.

ANTONIO FRANCESCO GENTILE

Antonio Francesco Gentile lives in Italy, Calabria, and is a software and network engineer. He works for a company in Rome as a network manager, with the “Culture Lab” http://culture.deis.unical.it Department of Telematics at University of Calabria, the computer science associations “Hacklab Cosenza” http://hacklab.cosenzainrete.it/ and “Verde Binario” http://www.verde- binario.org/ and is a freelance columnist for Italian magazines “Linux&C” http://www.oltrelinux.com/ and “Linux Magazine” http://www.linux-magazine.it/.

Hakin9

Dear BSD Readers! Currently Hakin9 creates publications concerning Unix area, and therefore, this is a good moment to share it with you. Now you can **download** both **Kernel and Wireshark** publications **for FREE** and implement our technical content directly into your computer. To get it, write your Hakin9 username (after registration on free account) in the message topic to **en@hakin9.org**, and next we will send you 2 magazines.
Don't hesitate!

www.hakin9.org

Hardening FreeBSD

with TrustedBSD and Mandatory Access Controls (Part 4)

Most system administrators understand the need to lock down permissions for files and applications. In addition to these configuration options on FreeBSD, there are features provided by TrustedBSD that add additional layers of specific security controls to fine tune the operating system for multilevel security.

What you will learn…

- Configuration of the mac_seeotheruids module.

What you should know…

- Basic FreeBSD knowledge to navigate the command line
- Familiarity with loader.conf to enable kernel modules at boot

Since version 5.0 of FreeBSD, the TrustedBSD extensions have been included with the default install of the operating system. By default, this functionality is disabled and requires support to be compiled in or kernel modules to be loaded at boot time. For the purpose of this article, support will be loaded in with kernel modules already available with FreeBSD 9. Part 4 of the TrustedBSD series will cover the basic configuration of the mac_seeotheruids module.

Warning

Incorrect MAC settings can cause even the root user to not be able to login to the system. Be sure to run these tests on a VM or test machine to avoid any issues with production systems. This article assumes that a fresh install of FreeBSD 9.0 has been performed before continuing.

As in the previous articles, a certain set of users will help to illustrate how to use *mandatory access controls* (MAC). For the `mac_seeotheruids` module, the purpose is

Listing 1. *Users setup on FreeBSD*

```
# pw user add -n user1 -s /bin/csh -m
# pw user add -n user2 -s /bin/csh -m
# pw user add -n user3 -s /bin/csh -m
# pw user mod -g user-reg -n user1
# pw user mod -g user-reg -n user2
# passwd user1
Changing local password for user1
New Password:
Retype New Password:
# passwd user2
Changing local password for user2
New Password:
Retype New Password:
# passwd user3
Changing local password for user3
New Password:
Retype New Password:
# groups user1
user1 user-reg
# groups user2
user2 user-reg
# groups user3
user3
# sysctl security.bsd.see_other_uids
security.bsd.see_other_uids: 1
# sysctl security.bsd.see_other_gids
security.bsd.see_other_gids: 1
```

to restrict the ability to view the running processes of other users on the system. Listing 1 shows a basic setup for the required users for this article.

The sysctl values for `security.bsd.see_other_uids` and `security.bsd.see_other_gids` are set to 1 to allow all users and groups to see the running processes of other users and groups on the system. The `mac_seeotheruids` module adds additional features beyond the normal `security.bsd.see_other_uids` and `security.bsd.see_other_gids` sysctl values on the system. Listing 2 shows how to load

Figure 1. *Using tmux and multiple window panes*

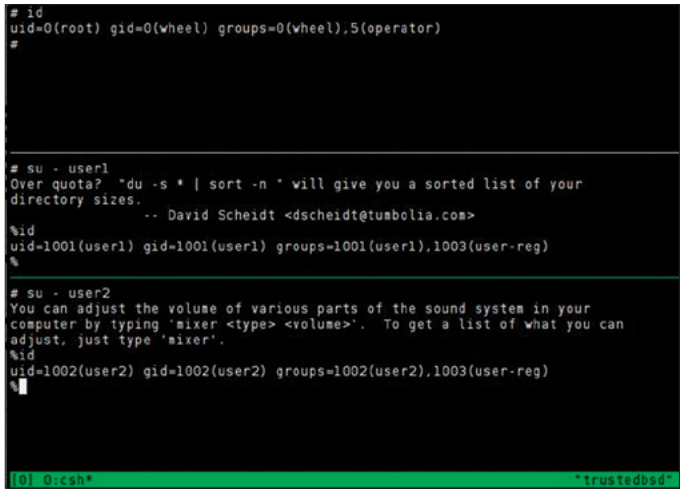


Figure 2. *Three separate window panes in tmux for the root, user1 and user2 users*

Listing 2. *Loading the mac_seeotheruids module on system startup*

```
# echo 'mac_seeotheruids_load="YES"' >> /boot/loader.conf
# echo 'security.mac.seeotheruids.enabled=0' >> /etc/sysctl.conf
# reboot
(The sysctl values change the default values which enables the module.)
```

Listing 3. *Installing tmux*

```
# pkg_add -r tmux
Fetching ftp://ftp.freebsd.org/pub/FreeBSD/ports/amd64/packages-9.0-release/Latest/tmux.tbz... Done.
Fetching ftp://ftp.freebsd.org/pub/FreeBSD/ports/amd64/packages-9.0-release/All/libevent-1.4.14b_2.tbz... Done.
```

Listing 4. *Start a loop in the user2 window then navigate to the user1 window and run "ps -aux|grep user2" to view the running processes*

```
(Type the following in the user2 window:)
%echo "while 1; echo 'user2' && sleep 1; end" | csh -f
user2
user2
..
(Use Ctrl-b and the arrow keys to go to the user1 window and type the following)
%ps aux | grep '^user2'
user1 88500   0.0  0.3    556    304   2  R+    7:12PM    0:00.00 grep user2
root  84292   0.0  1.6  41296   1828   3  I     6:45PM    0:00.01 su - user2
user2 84293   0.0  2.1  14612   2388   3 i S     6:45PM    0:01.21 -su (csh)
user2 88467   0.0  1.6  14612   1804   3  S+    7:12PM    0:00.01 csh -f
user2 88498   0.0  0.7   3916    780   3  S+    7:12PM    0:00.00 sleep 1
%
(user1 can see user2 processes)
```

Listing 5. *Set the sysctl value for security.bsd.see_other_uids to 0, then try to view the processes of user2 with user1. (Note: the while loop in the user2 window should still be echoing "user2")*

```
(In the root window, type the following)
# sysctl security.bsd.see_other_uids=0
security.bsd.see_other_uids: 1 -> 0
# ps aux | grep '^user2'
root  89049   0.0  0.3    556    308   1  R+    7:21PM    0:00.00 grep user2
root  84292   0.0  1.6  41296   1828   3  I     6:45PM    0:00.01 su - user2
user2 84293   0.0  2.1  14612   2388   3  I     6:45PM    0:01.21 -su (csh)
user2 88467   0.0  1.6  14612   1804   3  S+    7:12PM    0:00.25 csh -f
user2 89047   0.0  0.7   3916    780   3  S+    7:21PM    0:00.00 sleep 1

(Use Ctrl-b and the arrow keys to move to the user1 window and type the following)
%ps aux | grep '^user2'
%
(user1 can no longer view the running processes of the other users.)
```

Listing 6. *Set the sysctl value for security.mac.seeotheruids.primarygroup_enabled to 1, then try to view the processes of user2 with user1. (Note: the while loop in the user2 window should still be echoing "user2")*

```
(In the root window, type the following)
# sysctl security.mac.seeotheruids.primarygroup_enabled=1
security.mac.seeotheruids.primarygroup_enabled: 0 -> 1
# sysctl -a | grep security.mac.see
security.mac.seeotheruids.specificgid: 0
security.mac.seeotheruids.specificgid_enabled: 0
security.mac.seeotheruids.suser_privileged: 1
security.mac.seeotheruids.primarygroup_enabled: 1
security.mac.seeotheruids.enabled: 1

(Login as user3 in the root window and note that this user cannot see the processes of user2.)

# su - user3
%ps aux|grep '^user2'
%
%exit

(Use Ctrl-b and the arrow keys to move to the user1 window and type the following)

%ps aux | grep '^user2'
root   7871   0.0  1.5  41296   1652   3  I     9:26PM 0:00.01 su - user2
user2  7872   0.0  2.1  14612   2324   3  I     9:26PM 0:00.01 -su (csh)
user2  7876   0.0  1.6  14612   1812   3  S+    9:26PM 0:22.42 csh -f
user2 58995   0.0  0.7   3916    776   3  S+   11:16AM 0:00.00 sleep 1
%ps aux
USER    PID  %CPU %MEM    VSZ    RSS  TT  STAT STARTED    TIME COMMAND
root   7840   0.0  1.5  41296   1648   2  I    Mon09PM 0:00.01 su - user1
user1  7841   0.0  2.2  14612   2456   2  S    Mon09PM 0:00.05 -su (csh)
user1 45557   0.0  1.1  14328   1212   2  R+   10:43AM 0:00.00 ps aux
root   7871   0.0  1.5  41296   1648   3  I    Mon09PM 0:00.01 su - user2
user2  7872   0.0  2.1  14612   2324   3  I    Mon09PM 0:00.01 -su (csh)
user2  7876   0.0  1.6  14612   1812   3  S+   Mon09PM 0:59.99 csh -f
user2 45556   0.0  0.7   3916    776   3  S+   10:43AM 0:00.00 sleep 1
%

(user1 can now view the running processes of user2 and any processes that belong to the user-reg group.)
```

the module at startup with the use of `/boot/loader.conf`. In order to make it easier to monitor different users on the system, `tmux` will be installed to move between different user screens. The basic usage of `tmux` will be covered with additional information provided in the reference section. Listing 3 shows the installation steps for `tmux` using the binary package. Once `tmux` is installed, type `tmux` to invoke the terminal multiplexer and type the following:

Listing 7. *Set the sysctl value for security.mac.seeotheruids.specificgid to 1003, which is the gid for the user-reg group then try to view the processes of user2 with user1. (Note: the while loop in the user2 window should still be echoing "user2")*

```
(In the root window, type the following)

# sysctl security.mac.seeotheruids.primarygroup_enabled=0
security.mac.seeotheruids.primarygroup_enabled: 1 -> 0
# sysctl security.mac.seeotheruids.specificgid=1003
security.mac.seeotheruids.specificgid: 0 -> 1003
# sysctl security.mac.seeotheruids.specificgid_enabled=1
security.mac.seeotheruids.specificgid_enabled: 0 -> 1
#
(Login as user3 in the root window and note that this user cannot see the processes of user2 or any other user.)

# su - user3
%ps aux
USER    PID  %CPU %MEM    VSZ    RSS  TT  STAT STARTED    TIME COMMAND
user3 43979   0.0  2.1  14612   2340   1  S    10:18AM 0:00.02 -su (csh)
user3 43983   0.0  1.1  14328   1208   1  R+   10:18AM 0:00.00 ps aux
%ps aux | grep '^user2'
%
%exit

(Use Ctrl-b and the arrow keys to move to the user1 window and type the following)

%ps aux | grep '^user2'
user2  7872   0.0  2.1  14612   2324   3  I    Mon09PM    0:00.01 -su (csh)
user2  7876   0.0  1.6  14612   1812   3  S+   Mon09PM    0:59.53 csh -f
user2 44519   0.0  0.7   3916    776   3  S+   10:26AM    0:00.00 sleep 1
%ps aux
root   1126   0.0  1.6  18884   1832   0  I+   Mon07PM    0:00.01 tmux: client (/tmp/tmux-0/default) (tmux)
root   1129   0.0  2.1  14612   2396   1  Is   Mon07PM    0:00.09 -csh (csh)
root  43977   0.0  1.6  41296   1808   1  I    10:18AM    0:00.01 su - user3
user3 43979   0.0  2.1  14612   2340   1  I+   10:18AM    0:00.02 -su (csh)
root   1153   0.0  2.0  14612   2296   2  Is   Mon07PM    0:00.01 -csh (csh)
root   7840   0.0  1.5  41296   1648   2  I    Mon09PM    0:00.01 su - user1
user1  7841   0.0  2.2  14612   2456   2  S    Mon09PM    0:00.04 -su (csh)
user1 44709   0.0  1.1  14328   1260   2  R+   10:29AM    0:00.00 ps aux
root   1155   0.0  2.1  14612   2304   3  Is   Mon07PM    0:00.02 -csh (csh)
root   7871   0.0  1.5  41296   1648   3  I    Mon09PM    0:00.01 su - user2
user2  7872   0.0  2.1  14612   2324   3  I    Mon09PM    0:00.01 -su (csh)
user2  7876   0.0  1.6  14612   1812   3  S+   Mon09PM    0:59.61 csh -f
user2 44708   0.0  0.7   3916    776   3  S+   10:29AM    0:00.00 sleep 1
root   7163   0.0  2.3  14612   2616   4  Is+  Mon09PM    0:00.07 -csh (csh)

(user1 can now view all processes regardless of uid/gid.)
```

Listing 8. *Set the sysctl value for security.mac.seeotheruids.suser_privileged to 1, which restricts the root user from viewing*

```
(In the root window, type the following)
# sysctl security.mac.seeotheruids.suser_privileged=0
security.mac.seeotheruids.suser_privileged: 1 -> 0
#
# sysctl security.mac.seeotheruids.suser_privileged=0
security.mac.seeotheruids.suser_privileged: 1 -> 0
# ps aux|grep '^user2'
#
# sysctl security.mac.seeotheruids.suser_privileged=1
security.mac.seeotheruids.suser_privileged: 0 -> 1
# ps aux | grep '^user2'
user2  7872   0.0  2.1  14612   2324   3  I    Mon09PM    0:00.01 -su (csh)
user2  7876   0.0  1.6  14612   1812   3  S+   Mon09PM    1:00.29 csh -f
user2 46248   0.0  0.7   3916    776   3  S+   10:54AM    0:00.00 sleep 1
#
```

- Ctrl-b ”
- Ctrl-b ”
- Ctrl-b ”

This will open three window panes in one terminal, as seen in Figure 1.

Using Ctrl-b and the up and down arrow keys allows for the movement between window panes. Move to the top window, and run `id` then move to the middle pane and run `su - user1` then `id`. Move to the bottom pane and run `su - user2` then run `id`. The screen should look similar to that of Figure 2.

From this point on, the top, middle and bottom window panes will be referred to as the root, `user1` and `user2` windows. Listing 4 shows how `user1` is able to view the running processes of `user2`.

Both `user1` and `user2` are in the same user-reg group. The next step is to turn off the ability to see processes from others users and run the same test again. Listing 5 shows the steps necessary to run this test.

When loading the MAC module, it doesn't matter if the `security.bsd.see_other_uids` is set, as it is overridden by `secusecurity.mac.seeotheruids.enabled`. To allow members of a primary group to see their processes, the `security.mac.seeotheruids.primarygroup_enabled` value must be set to 1. Listing 6 shows the setting of this value which will allow `user1` to view the processes in its primary group.

Another feature of this module is the ability to exempt a group from the restrictions of the uid/gid policy. Listing 7 shows how to exempt the user-reg group from this policy, which will allow `user1` and `user2` to view all processes on the system.

References

- FreeBSD Handbook – Mandatory Access Cotnrol: *http://www.freebsd.org/doc/handbook/mac.html*
- MAC seeotheruids Module: *http://www.freebsd.org/doc/handbook/mac-seeotheruids.html*
- Mandatory Access Control: *http://en.wikipedia.org/wiki/Mandatory_access_control*
- Tmux: *http://tmux.sourceforge.net/*
- TrustedBSD: *http://www.trustedbsd.org/*

An additional sysctl value can actually restrict the root user from viewing the running processes of other users. Listing 8 shows the steps necessary to use the feature.

The examples in this article highlight how to separate the viewing of running processes by users in addition to the default features included with FreeBSD. The `security.bsd.see_other_uids sysctl` value can be enabled in the default sysctl.conf to provide some user separation as well as the default configurations for the `mac_seeotheruidsuids` modules. In later articles, the MAC modules will be combined to present different layers of security and to help with classifying information.

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.

SPTechCon

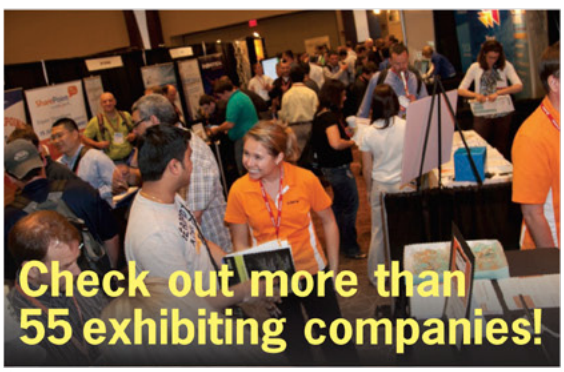
The SharePoint Technology Conference

March 3-6, 2013 → San Francisco

Get the scoop on SharePoint 2013!

Register Early and SAVE!

The Best SharePoint Training!



Choose from over 90 Classes & Workshops!

Check out these NEW! classes, taught by the industry's best experts!

- How to Install SharePoint 2013 Without Screwing It Up — Todd Klindt and Shane Young
- What IS SharePoint Development? — Mark Rackley
- SharePoint Performance: Best Practices from the Field — Jason Himmelstein
- Creating a Great User Experience in SharePoint — Marc Anderson
- Ten Best SharePoint Features You've Never Used — Christian Buckley
- Understanding and Implementing Governance for SharePoint 2010 — Bill English
- Building Apps for SharePoint 2013 — Andrew Connell
- SharePoint Solutions with SPServices — Marc Anderson
- Lists: Used, Abused and Underappreciated — Wes Preston
- Planning and Configuring Extranets in SharePoint 2010 — Geoff Varosky
- Creating Simple Dashboards Using Out-of-the-Box Web Parts — Jennifer Mason
- Integrating SharePoint 2010 and Visual Studio Lightswitch — Rob Windsor
- Solving Enterprise Search Challenges with SharePoint 2010 — Matthew McDermott
- Getting Stuff Done! Managing Tasks with SharePoint Designer Workflows — Chris Beckett
- SharePoint 2013 Upgrade Planning for the End User: What You Need to Know — Richard Harbridge
- Ten Non-SharePoint Technical Issues That Can Doom Your Implementation — Robert Bogue
- SharePoint MoneyBall: The Art of Winning the SharePoint Metrics Game — Susan Hanley
- Intro to Branding SharePoint 2010 in the Farm and Online — Randy Drisgill and John Ross
- How to Best Develop Requirements for SharePoint Projects — Dux Raymond Sy

Lots more online!

A BZ Media Event

CENTRAL CONTRACTOR REGISTRATION

Follow us: twitter.com/SPTechCon

www.sptechcon.com

EuroBSDcon and MeetBSD California

Two Continents One Community

MeetBSD Group Photo

This year's EuroBSDcon and MeetBSD California took place just a few weeks apart in two very different locations but together demonstrated seamless solidarity on the part of the BSD community. MeetBSD in Sunnyvale, California was like a reunion for many speakers and attendees who had recently met in Warsaw, Poland for EuroBSDcon. Some familiar European faces such as Robert Watson and Alexander Motin even made appearances only at the more distant event, showing once again that the geography of BSD and its community is "the Internet."

MeetBSD Group Bump!


Pawel Jakub Dawidek opening EuroBSDcon with a lesson about Polish histor

Marking its 11th year, EuroBSDcon 2012 chartered new territory by being the first of the series to take place in "New Europe", a decision that brought only novelty rather than discomfort. Should you choose, you could easily find

MeetBSD 2012 at Yahoo!

Starbucks, Subway, McDonald's and KFC in both cities, not to mention overall great food and shopping. The only surprise was the thick fog that complicated a few departures from Warsaw.


Kirk McKusick presenting his keynote, An Overview of Locking in the FreeBSD Kernel

This year's EuroBSDcon also marked a new milestone as being the first client event of the EuroBSDcon Foundation, a Dutch Stichting that exists to provide legal and financial infrastructure for the migratory conference. This BSD-agnostic body made a distinct impression on the event's program by ensuring near-equal representation of the leading BSD projects. A mild controversy even surrounded the rejection of several OpenBSD proposals due to sheer quantity. Who would have thought?


(No real comment – people watching the closing sessions)

The FreeBSD Developer Summits that preceded both events covered many of the usual topics like the toolchain and ports but Alistair Crooks wowed people with a Netflix presentation in Warsaw and I was happy to see the BHyVe hypervisor get strong attention in Sunnyvale. Scott Long from Netflix continued Alistair's message at MeetBSD with demonstration hardware to boot: Netflix is currently serving over 30% of the traffic on the Internet and is moving to an elegant, high-density server that can cache its content at ISPs around the world. Remarkably, the solution is basically a FreeBSD 9.1 web server that distributes several terabytes of video files from UFS. The solution is very current and very off-the-shelf. The big news for


The Closing Session

BHyVe is that everyone present agreed that it should be merged into the FreeBSD tree as soon as the developers see fit.

Brooks Davis presiding over the Toolchain session at the FreeBSD Developer Summit in Warsaw

Alistair Crooks shows the new Netflix server design to several developers at MeetBSD including Kris Moore

BHyVe developers Peter Grehan and Neel Natu at MeetBSD

Talk highlights included OpenBSD developer Philip Gunther's plentiful giving of credit where credit was due on the part of FreeBSD developers, during which Kirk McKusick said, "yep, you found a bug we need to fix in FreeBSD." Martin Matuska's talk about FreeBSD ZFS profiling and tuning using tools like `/usr/ports/sysutils/zfs-stats/` was also very good, especially considering how little attention these tools have received. John Hixson's FreeNAS system architecture talk provided a nice peek into how FreeNAS works under the hood, a topic that has also received little attention. Hopefully the videos for all of these will be online soon.

What the EuroBSDcon talks offered in breadth, the MeetBSD talks offered in depth. Adrian Chadd took several opportunities to hammer home the point that embedded FreeBSD has made huge progress in recent months and that several near-Tier 1 platforms are available here and now. He highlighted the need for a FreeBSD cross-compilation environment not unlike NetBSD's `build.sh`. Being a hot topic, embedded FreeBSD was presented as a talk, a full-group discussion and as a dedicated breakout session. At first this arrangement seemed like a deviation from the UnConference format but it turned out to be very effective in refining the discussion.

Most of the MeetBSD presentations are online at *https://www.meetbsd.com/conference/talks-and-sessions* and the EuroBSDcon ones should be up soon.

Did you miss out? Probably, but perhaps you are not aware that various travel grants exist for events like these. The organizers of EuroBSDcon, MeetBSD, AsiaBSDCon, BSDCan and NYCBSDCon all have travel grants available for presenters and the FreeBSD Foundation has helped dozens of people attend various events over the years, including developers from sympathetic projects. Google also offered financial support for female computer scientists to attend EuroBSDcon. Furthermore, I can safely say from personal experience that there was a time that each and every presenter could have never pictured themselves giving a talk at a conference. Events like EuroBSDon and MeetBSD are the heartbeat of the BSD community and I encourage you to find the time to attend one, submit a proposal or organize a BSD User Group in your area.

See you at the next BSDCon!

MICHAEL

Michael has used BSD Unix systems since 1991 and is the Editor of the BSD technical journal Call For Testing.

PgDay.IT 2012

The sixth edition of the Italian PostgreSQL Day (PgDay) held at the Monash University Center in Prato, Tuscany, on November the 23th has been a success. The Italian community did respond very well to the event, and guests from all over the country came to discuss, acquire knowledge and share experience about this great database.

The whole staff of the Italian PostgreSQL Users' Group (ITPUG) is proud of how smooth the sixth edition of the Italian PgDay, the national event dedicated to the PostgreSQL database, has been. It was a great event, with a lot of attendees from all over the country and every detail was simply perfect thanks to the effort of all the volunteers and ITPUG members who donated their time and effort to the organization of the event. And it was not easy: even though ITPUG has been organizing PgDay year after year, and it even handled the first European PgDay back in 2008, scheduling and running such an event is not a simple task. Luckily, most of this year's organizers did participate in the organization of previous events and therefore shared a common experience on tasks that needed to be done in order to make every participant feel comfortable.

As in the previous edition, the conference was held at the Monash University Center in Prato, Tuscany, in a great

Group picture of the Italian PgDay 2012

building where two rooms, the Grollo and Veneziana, were prepared with appropriate devices for each speech.

It is worth noting that, even if the name "PgDay" sounds like a "one-day" event, this is not the truth about the 2012 PgDay (and previous events too). In fact, the community gathered the evening before in a local pub to enjoy a "Pg-Beer" offered by one of the conference sponsors. It was the perfect place to meet other professionals and passionates and share some experiences and laugh at scary and strange stories (all based on database tales, of course!). The evening continued in a local restaurant, where the participants enjoyed a delicious Fiorentina steak, a very famous kind of meal in Tuscany. Again, a perfect way to sit down and talk to other people, all peers, and share experiences, opinions, tips and tricks and so on.

The day after, of course, the conference took place with the opening session from the ITPUG's president Gabriele Bartolini. The attendees then listened in a kind of religious respect to the keynote talk by Simon Riggs and Andres Freund that explained the ongoing work for multi-master replication, a feature that will take several releases to get fully implemented in PostgreSQL and that will make another giant step for this database in the race to be the leader on the SQL market.

A short coffee break served along with local pastry, and then two parallel sessions began. The Veneziana room was dedicated to tutorial sessions, with two introductory sessions from yours truly and two sessions on the development of stored procedures. In the meantime, in the Grollo room it was time to introduce new features coming for free with the current 9.2 release, migration from Oracle to PostgreSQL and techniques to monitor and keep PostgreSQL instances healthy.

Of course time flies when you have a program full of such talks, and a lot of attendees were literally jumping from one room to the other in order to get even a single bit of information...and then it was time for lunch. The buffet lunch was another demonstration of the Tuscany superiority when taking down to meal. And it was really nice to walk around and see a lot of people not only enjoying great food, but again talking and sharing experiences, providing each other suggestions to solve some specific problems, and so on. The atmosphere was really relaxed and there was time to joke around and even take a group picture with all the members of the staff and all the attendees.

The afternoon was again filled with two parallel sessions: the Veneziana room focused on database development with talks about database unit testing, log analysis and database design for high volumes of data. In the other room, experiences and case studies related to Java EE and the adoption of PostgreSQL for High Availability solutions in Italian health-care (two talks) captured the attention.

As a tradition so far, approaching the end of the event there was a unique session of lightning talks, talks that can be no longer than 5 minutes and that can be on almost any subject, idea, claim, consideration, experience, and so on related to the PostgreSQL (or the database in general) world.

Two lucky attendees won a signed copy of the latest PostgreSQL books, donated by one of the event sponsors. The conference ended on time, with the closing session and a recap of the day by the author.

And as years before, while the official event was at the end, the community one was not. In fact, no more than 20 minutes later, the PostgreSQL addicted were populating the nearby pub drinking another great PgBeer offered again from a conference sponsor.

I'd like to thank all the organizers for their great and professional activity, as well as all the sponsors, and all the speakers for their quality contributions, but most notably every single attendee for trusting in ITPUG and the PgDay and for letting PostgreSQL be such a great product.

See you at PgDay.IT 2013!

On the Web

Italian PostgreSQL Users' Group (ITPUG): *http://www.itpug.org*
PgDay.IT 2012: *http://2012.pgday.it*
PostgreSQL: *http://www.postgresql.org*

PgDay.IT 2012 by numbers

At the 2012 Italian PgDay there were 95 attendees, including 8 regular speakers, and a few lightning talk speakers. Seventy-eight percent of the attendees came from northern Italy, including Emilia Romagna at its edge, while the rest came from central and southern Italy. Fifteen regular talks were given during the day, including the technical keynote. The conference was organized with the help of two gold sponsors and one bronze, and the patronage of the local city and a university Open Source laboratory. The on-site dedicated staff was made up of 7 volunteers, and other ITPUG members joined the staff on demand.

LUCA FERRARI

Luca Ferrari lives in Italy with his wife and son. He is an Adjunct Professor at Nipissing University, Canada, a co-founder and the vice-president of the Italian PostgreSQL Users' Group (ITPUG). He simply loves the Open Source culture and refuses to log-in to non-Unix systems. He can be reached on line at http://fluca1978.blogspot.com.

Headquarters:
San Jose, CA

855.GREP.4.IX | Contact Us

99% Compatibility online now...

IXSYSTEMS AND YOU ARE THE PERFECT MATCH

SHARED INTERESTS

- [x] Enterprise Storage Solutions
- [x] Personalized Customer Service
- [x] Bold New Information Technology

I'm a: Storage Reseller

In: The EU

Looking for: Storage Solutions to Sell

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

Visit Today!

iXsystems

Technology Partner Seeking Resellers/Integrators for TrueNAS™ Storage Appliance

WWW.IXSYSTEMS.COM/PERFECTMATCH