BSD MAGAZINE

FOR NOVICE AND ADVANCED USERS

MeteorJS on FreeBSD 11

HOW TO BUILD A REALTIME CHAT WITH A FEW LINES OF CODE

MANAGE LARGE SOFTWARE DEVELOPMENT PROJECTS USING RCS TOOLS

WORKING WITH HAMMER FILE SYSTEM AND PFSES

HOW TO CREATE GUESTS FROM DIFFERENT OSES IN A FREEBSD HOST WITH BHYVE

GETTING TO GRIPS WITH THE GIMP

THE FREEBSD CRYPTIC CROSSWORD

VOL.8 NO.05
ISSUE 05/2014(58)
1898-9144

FREENAS MINI
STORAGE APPLIANCE

IT *SAVES* YOUR LIFE.

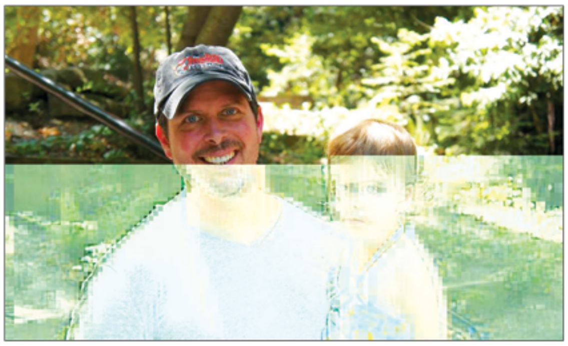
HOW IMPORTANT IS YOUR DATA?

Years of family photos. Your entire music and movie collection. Office documents you've put hours of work into. Backups for every computer you own. We ask again, *how important is your data?*

NOW IMAGINE LOSING IT ALL

Losing one bit - that's all it takes. One single bit, and your file is gone.

The worst part? **You won't know until you absolutely need that file again.**



Example of one-bit corruption

THE SOLUTION

The FreeNAS Mini has emerged as the clear choice to save your digital life. **No other NAS in its class offers ECC (error correcting code) memory and ZFS bitrot protection to ensure data always reaches disk without corruption and *never degrades over time*.**

No other NAS combines the inherent data integrity and security of the ZFS filesystem with fast on-disk encryption. No other NAS provides comparable power and flexibility. The FreeNAS Mini is, hands-down, the best home and small office storage appliance you can buy on the market. **When it comes to saving your important data, there simply is no other solution.**

The Mini boasts these state-of-the-art features:

- 8-core 2.4GHz Intel® Atom™ processor
- Up to 16TB of storage capacity
- 16GB of ECC memory (with the option to upgrade to 32GB)
- 2 x 1 Gigabit network controllers
- Remote management port (IPMI)
- Tool-less design; hot swappable drive trays
- FreeNAS installed and configured

iXsystems®

intel® inside™ ATOM™

http://www.iXsystems.com/mini

FREENAS CERTIFIED STORAGE

With over six million downloads, FreeNAS is undisputedly *the* most popular storage operating system in the world.

Sure, you could build your own FreeNAS system: research every hardware option, order all the parts, wait for everything to ship and arrive, vent at customer service because it *hasn't*, and finally build it yourself while hoping everything fits - only to install the software and discover that the system you spent *days* agonizing over **isn't even compatible**. Or...

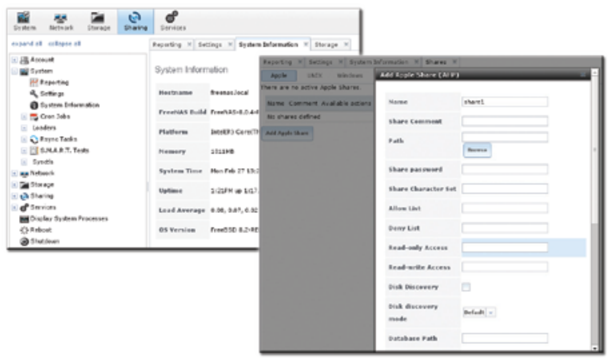
MAKE IT EASY ON YOURSELF

As the sponsors and lead developers of the FreeNAS project, iXsystems has combined over 20 years of hardware experience with our FreeNAS expertise to bring you FreeNAS Certified Storage. **We make it easy to enjoy all the benefits of FreeNAS without the headache of building, setting up, configuring, and supporting it yourself.** As one of the leaders in the storage industry, you know that you're getting the best combination of hardware designed for optimal performance with FreeNAS.

Every FreeNAS server we ship is...

» Custom built and optimized for your use case
» Installed, configured, tested, and guaranteed to work out of the box
» Supported by the Silicon Valley team that designed and built it
» Backed by a 3 years parts and labor limited warranty

As one of the leaders in the storage industry, you know that you're getting the best combination of hardware designed for optimal performance with FreeNAS. **Contact us today for a FREE Risk Elimination Consultation with one of our FreeNAS experts.** Remember, every purchase directly supports the FreeNAS project so we can continue adding features and improvements to the software for years to come. **And really - why would you buy a FreeNAS server from *anyone* else?**



System Information

Hostname	freenas.local
FreeNAS Build	FreeNAS-8.0.4-
Platform	Intel(R) Core(T
Memory	1011MB
System Time	Mon Feb 27 13:
Uptime	1:21PM up 1:17
Load Average	0.00, 0.07, 0.02
OS Version	FreeBSD 8.2-RE

There are no active Apple shares.

Name Comment Available actions
No shares defined

Add Apple Share (AFP)

Name	share1
Share Comment	
Path	
Share password	
Share Character Set	
Allow List	
Deny List	
Read-only Access	
Read-write Access	
Disk Discovery	☐
Disk discovery mode	Default
Database Path	

FreeNAS 1U

- Intel® Xeon® Processor E3-1200v2 Family
- Up to 16TB of storage capacity
- 16GB ECC memory (upgradable to 32GB)
- 2 x 10/100/1000 Gigabit Ethernet controllers
- Redundant power supply

FreeNAS 2U

- 2x Intel® Xeon® Processors E5-2600v2 Family
- Up to 48TB of storage capacity
- 32GB ECC memory (upgradable to 128GB)
- 4 x 1GbE Network interface (Onboard) - (Upgradable to 2 x 10 Gigabit Interface)
- Redundant Power Supply

intel inside™ XEON®

http://www.iXsystems.com/storage/freenas-certified-storage/

Dear Readers,

I hope you all are anxiously anticipating the May issue of BSD magazine. This time, we prepared very good articles that will cover many new topics. Just sit and read the new issue. This time I would like to make a very short presentation of what is included in the May issue. Today, you will learn:

- *How to manage large software development projects using RCS tools.*
- *How to use and administer RCS tools for BSD-Unix systems.*
- *What is DragonFly and how to use its most prominent file system Hammer.*
- *What is Meteor.*
- *What are some key features of MeteorJS.*
- *How to build a realtime chat with a few lines of code.*
- *How to understand bhyve, the BSD hypervisor.*

We would also like to thank you Authors, Reviewers, Proofreaders, BSD fans, Friends, and Readers for your invaluable support and contribution.

Enjoy your reading!
Ewa & BSD Team

MAGAZINE BSD

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

Contributing:
Michael Shirk, Andrey Vedikhin, Petr Topiarz, Charles Rapenne, Anton Borisov, Jeroen van Nieuwenhuizen, José B. Alós, Luke Marsden, Salih Khan, Arkadiusz Majewski, BEng, Toki Winter, Wesley Mouedine Assaby, Rob Somerville

Top Betatesters & Proofreaders:
Annie Zhang, Denise Ebery, Eric Geissinger, Luca Ferrari, Imad Soltani, Olaoluwa Omokanwaye, Radjis Mahangoe, Mani Kanth, Ben Milman, Mark VonFange

Special Thanks:
Annie Zhang
Denise Ebery

Art Director:
Ireneusz Pogroszewski

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

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

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

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

Publisher:
Hakin9 Media SK
02-676 Warsaw, Poland
Postepu 17D
Poland
worldwide publishing
editors@bsdmag.org
www.bsdmag.org

Hakin9 Media 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.

How to Manage Large Software Development Projects Using RCS Tools

Software Configuration Management has been a traditional concern since the beginnings of the IT era. BSD development teams have traditionally used CVS as the main version control system for all their projects, including kernel development. However, CVS has some major drawbacks that can be superseded using more modern tools that outclass CVS in order to achieve the same goal.

What you will learn…

- How to manage large software development projects using RCS tools
- Make the right choice depending on the nature of your programming project
- Understand the weaknesses and strengths of the different RCS paradigms in use
- How to use and administer these RCS tools for BSD-Unix systems

What you should know…

- Intermediate UNIX OS background as end-user and administrator
- Some experience with CVS environment for version control
- Experience with Ports system package
- Software development experience

One of the more popular VCS tools was a system called RCS, which is still distributed with many computers today. Even the popular Mac OS X operating system includes the `rcs` command when you install the Developer Tools. This tool basically works by keeping patch sets (that is, the differences between files) from one revision to another in a special format on disk; it can then recreate what any file looked like at any point in time by adding up all the patches.

Notice that there are other Open Source systems for RCS not using a change set approach but snapshots like Git, originally developed in C and Perl, which use SHA-1 digests instead of numbers to control the different versions stored in the distributed repositories.

Repository Models. Centralized vs. Distributed

The next major issue that people encounter is that they need to collaborate with developers on other systems. To deal with this problem, Centralized Version Control Systems (CVCSs) were developed. These systems, such as CVS and Subversion, have a single server that contains all the versioned files, and a number of clients that check out files from that central place. For many years, this has been the standard for version control (see Figure 1-2).

This setup offers many advantages, especially over local VCSs. For example, everyone knows to a certain degree what everyone else on the project is doing. Administrators have fine-grained control over who can do what; and it's far easier to administer a CVCS than it is to deal

Dr.WEB®
since 1992

Dr.Web 9.0
for Windows — the rapid response anti–virus

1. Reliable protection against the threats of tomorrow
2. Reliable protection against data loss
3. Secure communication, data transfer and Internet search

www.drweb.com

Free 30-day trial: https://download.drweb.com

New features in Dr.Web 9.0 for Windows: http://products.drweb.com/9

FREE bonus — Dr.Web Mobile Security:
https://download.drweb.com/android

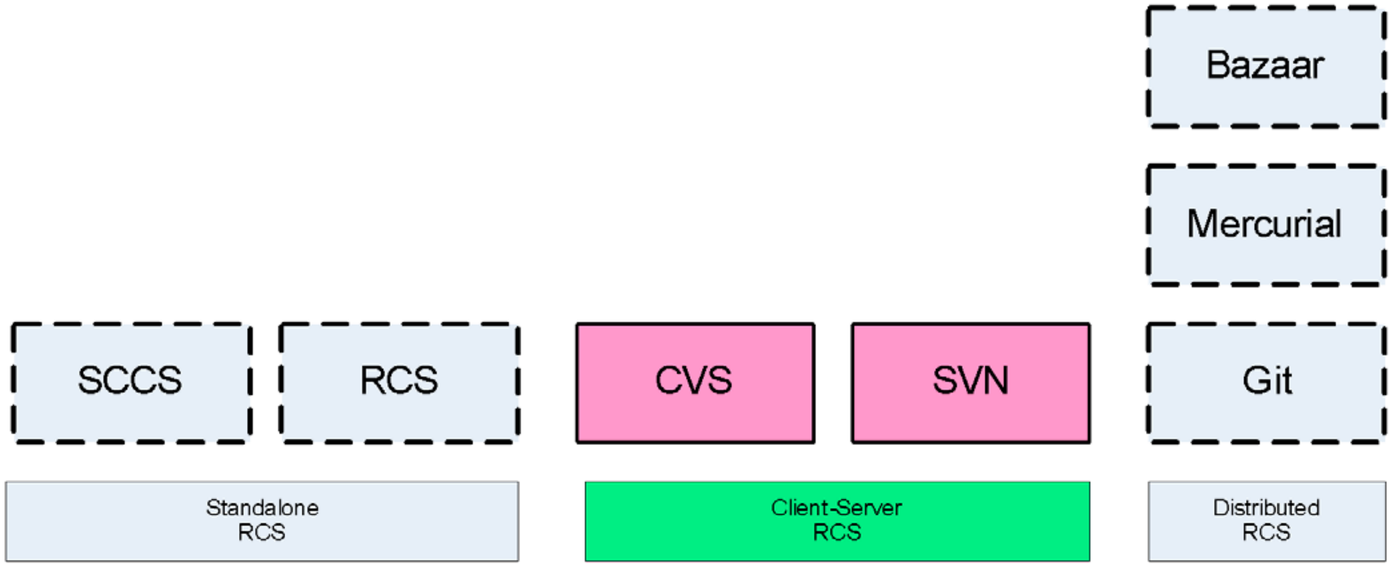


Figure 1. *Overview of current repository models and tools used in VCS*

with local databases on every client. However, this setup also has some serious downsides. The most obvious is the single point of failure that the centralized server represents. If that server goes down for an hour, then during that hour nobody can collaborate at all or save versioned changes to anything they're working on. If the hard disk the central database is on becomes corrupted, and proper backups haven't been kept, you lose absolutely everything—the entire history of the project except whatever single snapshots people happen to have on their local machines. Local VCS systems suffer from this same problem – whenever you have the entire history of the project in a single place, you risk losing everything. This is where Distributed Version Control Systems (DVCSs) step in. In a DVCS (such as Git, Mercurial, Bazaar or Darcs), clients don't just check out the latest snapshot of the files: they fully mirror the repository. Thus if any server dies, and these systems were collaborating via it, any of the client repositories can be copied back up to the server to restore it. Every checkout is really a full backup of all the data (see Figure 1-3).

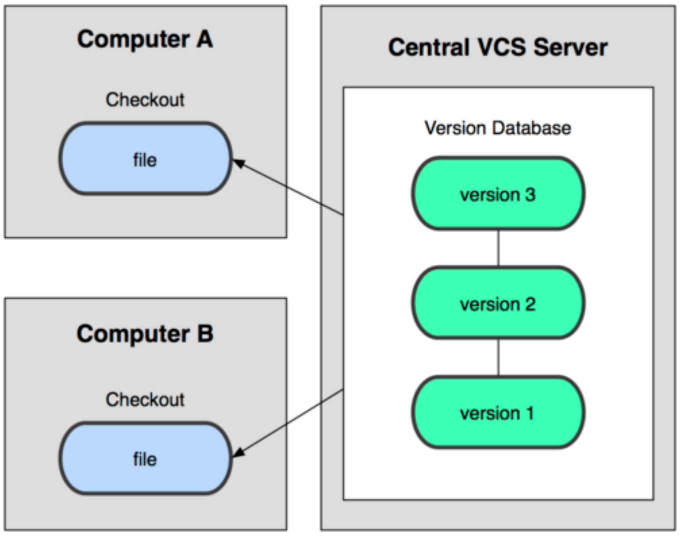


Figure 2. *Centralised repository model*

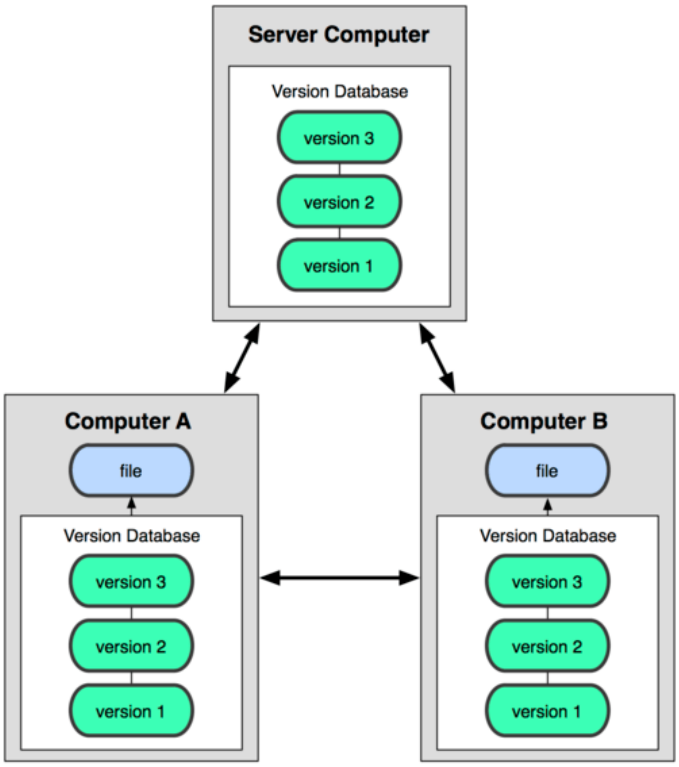


Figure 3. *Distributed repository model*

For this reason, one of the most important drawbacks for distributed repositories is the poor performance for initial repository loads. This is the price to be paid for higher availability and failure protection using this redundant approach.

SVN and git Internals

Snapshots versus differences

Traditionally, RCS systems have stored the different versions of a file by using differences with respect to its immediate predecessor or successor in time. Historically the first RCS systems stored all consecutive differences between a given version and its previous one: Figure 4.

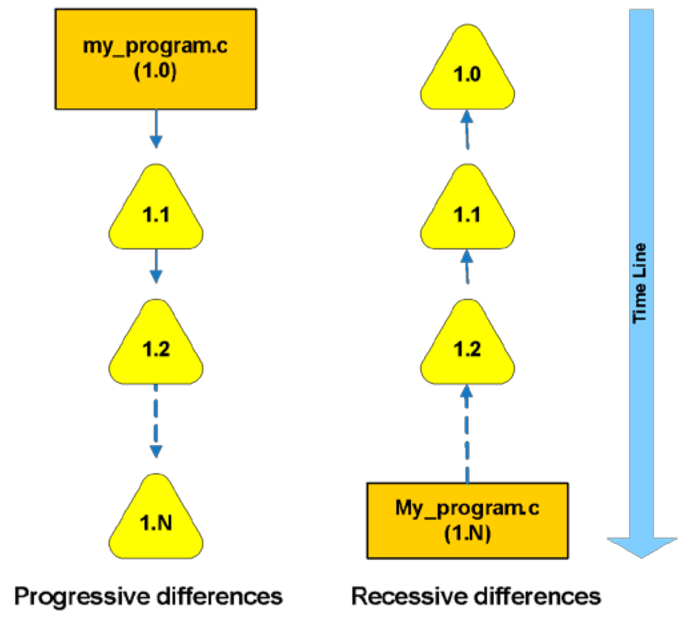


Figure 4. *RCS Differences Storage Systems (single file)*

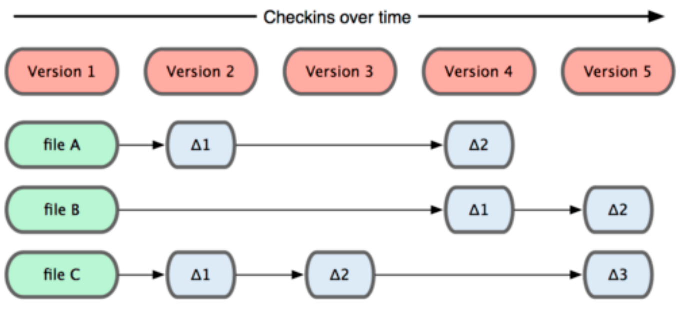


Figure 5. *Differences storage for RCS (CVS and SVN)*

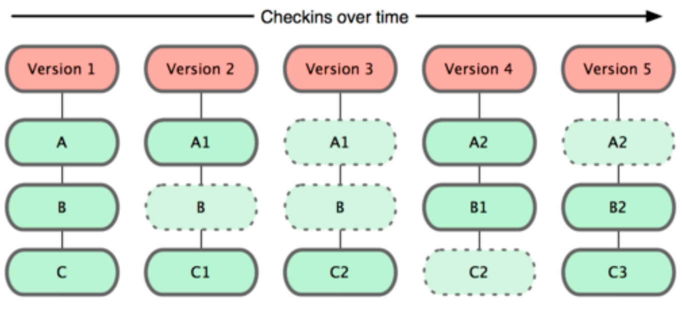


Figure 6. *Snapshots storage for RCS (Git and Mercurial)*

Obviously, the files belonging to a project most widely used are the latest ones released in time. This is the reason why the more evolved RCS systems, such as CVS and SVN among others, prefer the approach based upon storing recessive differences. Hence the way to get a previous version consists of applying on the latter file the previous differences stored until the desired version of such a file is reached (Figure 5). Furthermore, beyond SVN client-server approach, the systems like Git and Mercurial also allow the use of snapshots, which is a promising feature for developers shown in Figure 6.

Snapshots allow developers to get an accurate picture on a specific moment in time, which can be used later on to derive alternatives (variants) from the main development branch, and to get a frozen view of the current status of a project, suitable for review processes and audits.

Branching and Merging

Developers often need to maintain a release version of a product while working on new features. The base/release version of a project in Git/SVN is the *trunk*. New versions of the trunk are developed in branches off of the trunk or off of another branch. In Subversion, you work on branches in a new working copy of your code. Git maintains a single working copy and enables you to switch branches at will.

When changes are complete, and a branched version is ready to become the release version, it is merged with another branch (or the trunk) to create a unified code base. Conflicts are dealt with, and then the finished code is committed to the repository and pushed (with Git) to the remote server, if desired.

Distributed SCM Applications. Git

Before starting with this section, we assume the reader has a certain knowledge about the CVS client-server system, as the main aim of these paragraphs is to explain by comparing the different approaches and strategies followed by both SCM systems. Since the 80s CVS is the “de facto” revision control system used for most development projects, for instance BSD OS flavors.

Git is a VCS which allows complex workflows suitable for almost every development team depending on the complexity level required by a project:

In this sense, Git supports three workflows for VCS:

- Centralised workflow; this is the approach followed by RCS, CVS and SVN consisting of the traditional client-server paradigm.

- Flow workflow, useful when dealing with several teams devoted to a single project in which branches have been defined; for instance, for patching or alternate releases purposes.

The Gitflow Workflow streamlines the release cycle by using isolated branches for feature development, release preparation, and maintenance. Its strict branching model also lends some much needed structure to larger projects.

- Distributed workflow, also termed forking workflow, is the most powerful way to ensure reliability against single-failure repositories. This workflow takes full advantage of branching and cloning features provided by Git.

Summing up, the distributed workflow is the preferred approach when dealing with Git in order to provide a safe and reliable environment to manage large teams. It can also accept commits from untrusted individuals.

Git explained

Suppose you are involved in a large software project in which workload is not equally distributed around a developers community. In these cases, some features from SVN and CVS systems such as the use of numbers to label revisions or a centralized repository are not a good idea in terms of performance and availability and the right decision to make is to adopt a distributed workflow for VCS.

That is why, Git plays an important role for projects in which the workload is non-linear. The main characteristics for Git are:

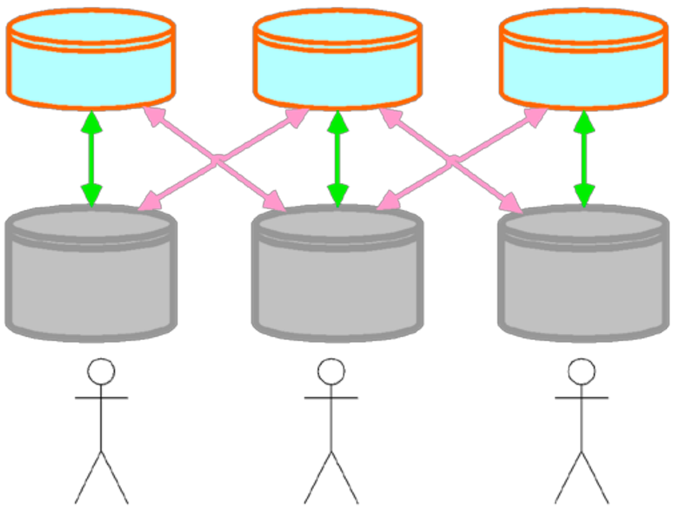


Figure 7. *Distributed (forking) workflow*

- Command syntax, easy to get familiar with all command-line git commands provided by Git. Besides, most of the commands exist also for the SVN platform:

```
$ git [ command ]
```

- Repositories. While SVN has a single repository for each project, Git has its own repository for each copy of a project.
- Uniform Resource Locators (URL). Every SVN repository counts with three directories named branches, tags and trunk and the versions are identified with numbers. For a distributed environment such as Git, this approach becomes unpractical and hence, Git uses 160-bit SHA1 digests to label project revisions.
- Commits. Every commit has an author and a field to record who/when created the change and who/when committed it to the repository.

An additional feature that makes Git especially wonderful for Open Source software developers is the ability of working with patches coming via email by using git format-patch and git send-email. Besides, Git also incorporates incorporates colourful output to provide better visualisation.

Git syntax and mostly used operations

The git(1) syntax is based on the getopt(1) UNIX paradigm, supporting a set of options for a given command and their required arguments:

```
git [--version] [--help] [-c name=value]
          [--exec-path[=<path>]] [--html-path] [--man-
   path]
          [--info-path]
          [-p|--paginate|--no-pager] [--no-replace-
   objects] [--bare]
          [--git-dir=<path>] [--work-tree=<path>]
          [--namespace=<name>]
          <command> [<args>]
```

The most commonly used git commands are:

- Add: Add file contents to the index
- bisect: Find by binary search the change that introduced a bug
- branch: List, create, or delete branches
- checkout: Checkout a branch or paths to the working tree
- clone: Clone a repository into a new directory
- commit: Record changes to the repository
- diff: Show changes between commits, commit and working tree, etc

Listing 1. *Creating a new Git repository (page 10/60)*

```
$ git config --global user.name "Jose B. Alos"
$ git config --global user.email jose.alos@yahoo.es
$ git init
$ git add .
$ git commit
Added new files to Git
# Please enter the commit message for your changes. Lines starting # with '#' will be ignored, and an empty message
   aborts the commit.
# On branch master
#
# Initial commit
#
# Changes to be committed:
#    (use "git rm --cached <file>..." to unstage)
#
#       new file:   bubble_sort.c
#       new file:   heapsort.c
#       new file:   pcp.sh
#       new file:   primes.cc
#
```

Listing 2. *Checking current status for Git repository (page 10/60)*

```
ecl51991:~/scripts/unix/scripts/unix> git status
# On branch master
#
# Initial commit
#
# Changes to be committed:
#    (use "git rm --cached <file>..." to unstage)
#
#       new file:   bubble_sort.c
#       new file:   heapsort.c
#       new file:   pcp.sh
#       new file:   primes.cc
#
```

Listing 3. *Cloning a Git repository (page 10/60)*

```
$ git clone /home/c20395/scripts/unix/scripts/unix my_cloned_proj
Cloning into 'my_cloned_proj...
The authenticity of host 'ecl51991. (10.128.40.37)' can't be established.
ECDSA key fingerprint is 6c:8d:f7:a1:dd:31:48:ed:05:ad:c8:97:75:6a:9e:bb.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ecl51991.,10.128.40.37' (ECDSA) to the list of known hosts.
c20395@ecl51991.'s password:
Cannot open display "default display"
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 6 (delta 0), reused 0 (delta 0) Receiving objects: 100% (6/6), done.
```

- fetch: Download objects and refs from another repository
- grep: Print lines matching a pattern
- init: Create an empty Git repository or reinitialize an existing one
- log: Show commit logs
- merge: Join two or more development histories together
- mv: Move or rename a file, a directory, or a symbolic link
- pull: Fetch from and integrate with another repository or a local branch
- push: Update remote refs along with associated objects
- rebase: Forward-port local commits to the updated upstream head
- reset: Reset current HEAD to the specified state
- rm: Remove files from the working tree and from the index
- show: Show various types of objects
- status: Show the working tree status
- tag: Create, list, delete or verify a tag object signed with GPG

In order to clarify the use for Git newcomers, the following sections will describe the most common operations to start with and manage project files under Git control.

Getting started with Git

Let us suppose we have installed all required Git components for our platform (BSD, Linux, SunOS, HP-UX, et al.) and that we have a set of files and directories corresponding to a new project.

- Creating a new Git repository from scratch. Once we are placed within the directory containing the project files, issue the following commands: Listing 1. After these files have been committed, we can check their statuses by running the command: Listing 2.
- Getting a copy of the project files for read-only purposes. If we want to examine some files belonging to our project under Git control, the following commands apply: Listing 3.

The changes performed can be committed to the original repository by using the pull subcommand:

```
$ git pull
```

Furthermore it is possible to add remote branches to a cloned repository by using the command:

```
$ git remote add remote url
$ git remote show remote
```

Eventually, the process of fetching and merging remote branches can be automated by using the git pull command, which is equivalent to the `svn update` command in SVN systems.

Strategies of GIT usage

Usage for Cooperative development

A widely used tactic in OpenSource projects that might be followed here is based upon the use of a CVS/SVN repository to provide updated working copies for developers. These copies can be modified later on and these modifications will be sent to the Git repository administrators in order to make the decision on including these in further versions.

To achieve this goal, the process to be executed can be summarized in the following steps, that can be extrapolated for Git usage in the same manner that we did for SVN.

In order to clarify its usage, consider two users named `userA` and `userB`, where userA is the Git's repository owner.

- Get a working copy of the repository. In the case of using Git, it is required to indicate the authentication process defined on this repository.
  - Local access to Git repository. Users share the same platform

```
$ git clone /home/userA/my_project my_cloned_project
```

Now the contents are available at my_cloned_project/ directory.

```
$ cd my_cloned_project
```

After some changes have been done, commit them into the Git directory:

```
$ git commit -a
```

And notify the other users of the changes so that the repository owner can update it by issuing the following commands:

```
$ cd /home/userA/my_project
$ git pull /home/userB/my_cloned_project master
```

The last command merges the changes from userB's master branch into userA's current branch repository. Keep in mind that any changes performed meanwhile by `userA` shall be fixed by hand.

Meet the **Developer-Friendly** Payment Solution

3 easy steps to optimized checkouts:

1

Create the checkout page

With Gate2Shop, you can optimize your payment pages by using ready-made templates or by customizing payment pages to your site look and feel.

2

Test and optimize

An effective payment page variant testing tool, A/B Testing helps you gain insight into user behaviour, increase payment conversion in the short and long term.

3

Accept payments worldwide

With dozens of alternative and local payment methods offered in multiple currencies, the personal-ized checkout allows you to reach users from all around the world.

✓ Easy integration ✓ Cross-platform ✓ Secure

G2S gate2shop Sell. More.

Call for a free consultation: +44 20 3051 0330
www.g2s.com

- Remote access to Git repository. Users do not share the same platform.

If we suppose the main Git repository is available at Persephone host, users working in other hosts can access the Git repository by using the SSH protocol, which is a built-in feature. Hence the unique change regarding Git local access is for the git clone subcommand:

```
    $ git clone persephone:/home/userA/my_project my_
    cloned_project
Cloning into ‘my_cloned_project’...
c20395@persephone.’s password:
Cannot open display “default display”
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 6 (delta 0), reused 0 (delta 0) Receiving
    objects: 100% (6/6), done.
```

Notice that the URL corresponding to a Git repository is stored under `.git/` directory and can be examined by issuing the command below. A single Git repository also supports variants for remote access based on SSH; it is also possible to use RSYNC or HTTP protocols for this purpose.

Usage for branching and merging development

A single Git repository supports also supports variant development by using branches, in order to meet the three basic requirements for complex-project management:

- Multiple project allocation
- Mainline versioning control supported
- Branching and variant development supported

Every Git repository counts with a default branch named “master”. Let’s suppose a working copy of a Git repository has been placed under our control. The process of using branches can be summarized in the following steps:

- Examine the existing branches for the project

```
$ git branch
```

  - Create a new branch labeled “experimental”

```
$ git branch experimental
```

  - Make the changes or additions/deletions desired
  - Commit the changes to the new “experimental” branch

```
$ git commit -a
```

The merging process in which a non-default branch must be incorporated into the “master” branch, can be easily performed by the command:

```
$ git merge experimental
```

Whenever the changes between both branches do not conflict with each other.

Basic Work Cycle

Git has numerous features, options, bells, and whistles, but on a day-to-day basis, odds are that you will use only a few of them. In this section, we’ll enumerate the most common things that you might find yourself doing with Git in the course of your daily work.

The typical work cycle looks like this:

- Update your working copy
  - *git clone*: Bring changes from the repository into the working copy.
- Make changes
  - *git add*: Put files and directories under version control, scheduling them for addition to the repository. They will be added in the next commit.
  - *git rm*: Remove files and directories from version control and from the local working directory tree.
  - *git fetch*: Duplicate something in working copy or repository, remembering history.
  - *git mv*: Move and/or rename something in working copy or repository.
- Examine your changes
  - *git status*: Print the status of working copy files and directories.
  - *git diff*: Display the differences between two revisions or paths.
- Fetch from and merge with another repository or local branch
  - *git pull*: Adds changes to another Git repository
- Resolve conflicts (merge others’ changes).
  - *git status*: Resolve conflicts on working copy files or directories.
- Commit your changes.
  - *git commit*: Send changes from your working copy to the repository.
- Set up a new label (optional)
  - *git tag*: Creates a label to reference a baseline (snapshot files)

BASIC OPERATIONS

This section will introduce the use of Git in a normal day’s work (how to get a working copy of the repository, deal with modifications and put all new stuff back into the repository). In order to provide a better understanding, let us assume we use a MS Windows client platform to operate with SVN server.

Git GUI Clients

The usage of the command-line gui tool is sometimes tricky for new users. That is the reason why a wide variety of tools to provide a better user interface for Git have been developed during the last few years.

Multiplatform GUI clients

There are two standard tools bundled with the Git environment, and developed by using the Tcl/Tk language. The first one is GITK, which is an application built using Tcl/Tk suitable for repository browsing purposes.

Figure 8 reflects the status of the Git project initially created while Figure 9 shows the repository status after he "experimental" branch has been created.

The second one is GIT-GUI, which goes beyond GITK and provides options to handle commits and branches management.

Figure 10 shows the changes made onto a file during the development phase in an intuitive way. For Unix-lovers, the tandem GITK/Git-View is good enough for a first approach to Git usage.

MS Windows GUI clients

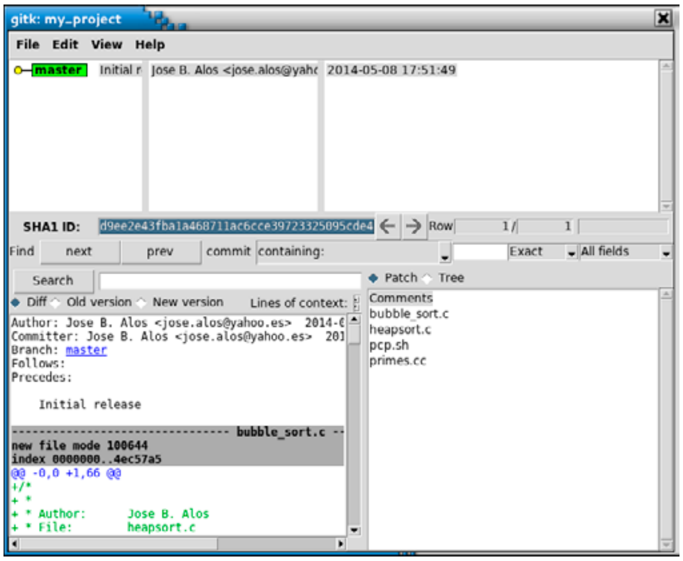
Unfortunately, we cannot avoid the extensive use of MS Windows platforms for serious development. For these users, two tools are available, which require the previous installation of Git 1.9.0 for MS Windows, available at *http://code.google.com*.

TortoiseGit, developed by Li Frank and inspired by well-known TortoiseCVS and TortoiseSVN, implements most regular features provided by Git such as commits, branches management, repository analysis and examination. The main drawback is that it is still under active development and the current stable version is 1.8.8.0 (Figure 11).

Git Extensions is the alternative and contains a standalone GUI together with a Visual Studio 2005/2008 plugin and is written in C# and C++. It is also not a mature alternative.

The most relevant features for these Git clients are:

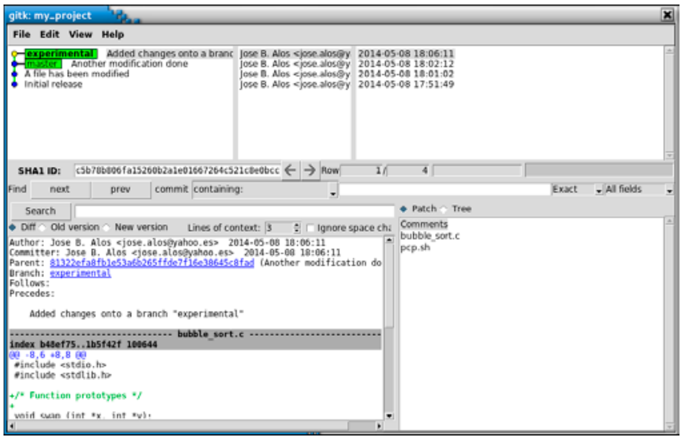
- Shell integration: TortoiseGit integrates seamlessly into the Windows shell (i.e. the explorer). This means you can keep working with the tools you're already familiar with. You do not need to change into a different application each time you need functions of the version control.
- Icon overlays: The status of every versioned file and folder is indicated by small overlay icons. That way you can see right away what the status of your working copy is.
- Easy access to Git commands: All Subversion commands are available from the explorer context menu. TortoiseGIT adds its own submenu there.


gitk: my_project
File Edit View Help
Find next prev commit containing: Exact All fields
Search
Patch Tree
Diff Old version New version Lines of context:

```
master  Initial r  Jose B. Alos <jose.alos@yahc  2014-05-08 17:51:49
SHA1 ID: d9ee2e43fba1a468711ac6cce39723325095cde4   Row 1/ 1

Author: Jose B. Alos <jose.alos@yahoo.es>  2014-(
Committer: Jose B. Alos <jose.alos@yahoo.es>  201
Branch: master
Follows:
Precedes:

    Initial release

-------------------------------- bubble_sort.c --
new file mode 100644
index 0000000..4ec57a5
@@ -0,0 +1,66 @@
+/*
+ *
+ * Author:      Jose B. Alos
+ * File:        heapsort.c
```

```
Comments
bubble_sort.c
heapsort.c
pcp.sh
primes.cc
```

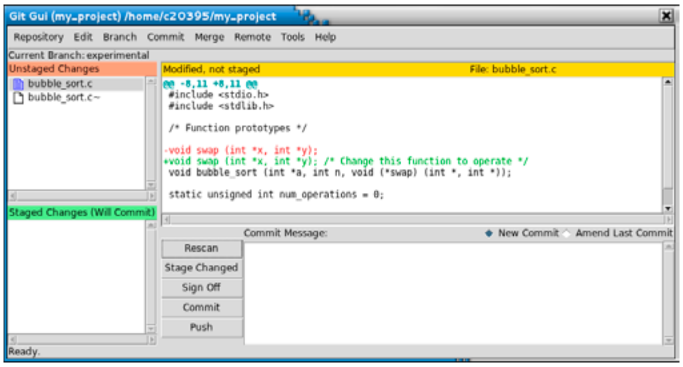
Figure 8. *GITK X11 client repository view (initial)*


gitk: my_project
File Edit View Help
Find next prev commit containing: Exact All fields
Search
Patch Tree
Diff Old version New version Lines of context: 3 Ignore space ch

```
experimental  Added changes onto a branc  Jose B. Alos <jose.alos@y  2014-05-08 18:06:11
master  Another modification done      Jose B. Alos <jose.alos@y  2014-05-08 18:02:12
A file has been modified               Jose B. Alos <jose.alos@y  2014-05-08 18:01:02
Initial release                        Jose B. Alos <jose.alos@y  2014-05-08 17:51:49
SHA1 ID: c5b78b806fa15260b2a1e01667264c521c8e0bcc   Row 1/ 4

Author: Jose B. Alos <jose.alos@yahoo.es>  2014-05-08 18:06:11
Committer: Jose B. Alos <jose.alos@yahoo.es>  2014-05-08 18:06:11
Parent: 81322efa8fb1e53a6b265ffde7f16e38645c8fad (Another modification do
Branch: experimental
Follows:
Precedes:

    Added changes onto a branch "experimental"

----------------------------- bubble_sort.c -----------------------------
index b48ef75..1b5f42f 100644
@@ -8,6 +8,8 @@
 #include <stdio.h>
 #include <stdlib.h>

+/* Function prototypes */
+
 void swap (int *x, int *y);
```

```
Comments
bubble_sort.c
pcp.sh
```

Figure 9. *GITK Repository view (with branch "experimental")*


Git Gui (my_project) /home/c20395/my_project
Repository Edit Branch Commit Merge Remote Tools Help
Unstaged Changes
Staged Changes (Will Commit)
Commit Message:
New Commit Amend Last Commit
Rescan
Stage Changed
Sign Off
Commit
Push

```
Current Branch: experimental
bubble_sort.c
bubble_sort.c~
Modified, not staged                                   File: bubble_sort.c
@@ -8,11 +8,11 @@
 #include <stdio.h>
 #include <stdlib.h>

 /* Function prototypes */

-void swap (int *x, int *y);
+void swap (int *x, int *y); /* Change this function to operate */
 void bubble_sort (int *a, int n, void (*swap) (int *, int *));

 static unsigned int num_operations = 0;
Ready.
```

Figure 10. *GIT-VIEW X11 Client*

- Graphical implementation of some Git commands: Such as diff or conflicts resolution.

TortoiseGIT also requires installation of Windows Installer 4.5 or later before attempting to install it for MS Windows XP SP3 systems or later.

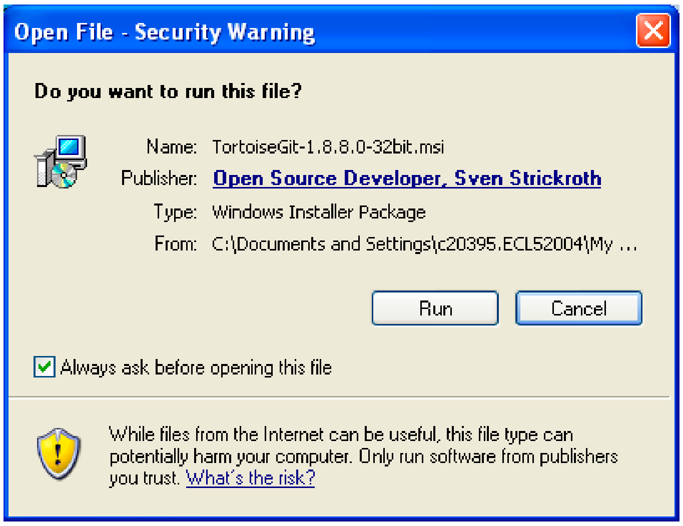
MacOS X GUI clients

For Apple lovers, the native Xcode development environment includes high-quality, built-in Git support.

Other interesting possibilities when using some free IDE such as Eclipse, is the use of Git plugins in a similar way as done with other VCS tools.

Browsing GIT Repositories

There are two main ways to browse the contents of a given Git repository. The first possibility, widely used by MS Windows-lovers is the use of the Git Tortoise client (RepoBrowser option), which requires previous authentication to access: Figure 12.



Do you want to run this file?

Name: TortoiseGit-1.8.8.0-32bit.msi
Publisher: Open Source Developer, Sven Strickroth
Type: Windows Installer Package
From: C:\Documents and Settings\c20395.ECL52004\My ...

[✓] Always ask before opening this file

While files from the Internet can be useful, this file type can potentially harm your computer. Only run software from publishers you trust. What's the risk?

Figure 11. *Git Tortoise client installation (page 14/60)*

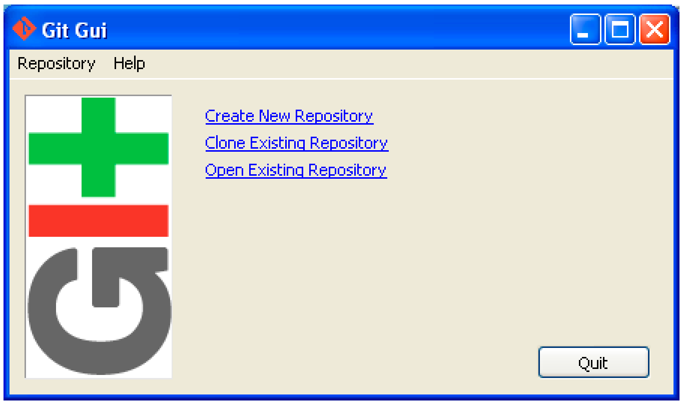


Figure 12. *Git repository browsing*

The common structure for controlling changes and their time evolution is given in Figure 13.

The second one is to use Tortoise Git support as shown in Figure 9. This is platform-independent and does not require an external tool, just a compliant HTTP browser such as MSIE or Mozilla among others.

Getting a working copy from Git repository

To start with Git clients, the first step is to get a copy of an empty repository to be populated with the desired directories structure for our development project (Figure 14).

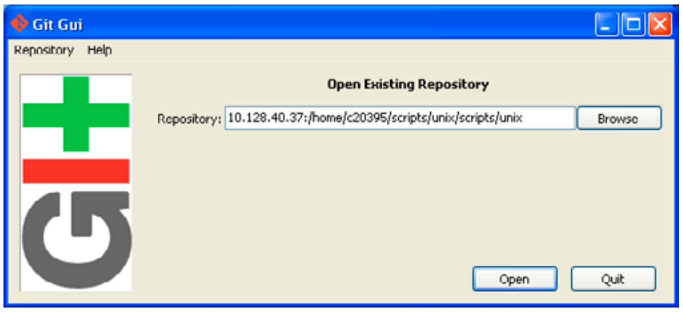
In UNIX-like systems, the equivalent commands are:

```
$ cd $HOME
$ git clone 10.128.40.37:/home/userA/(
```

If you only want to get a copy for read-only purposes, just use the subcommand `git fetch`.

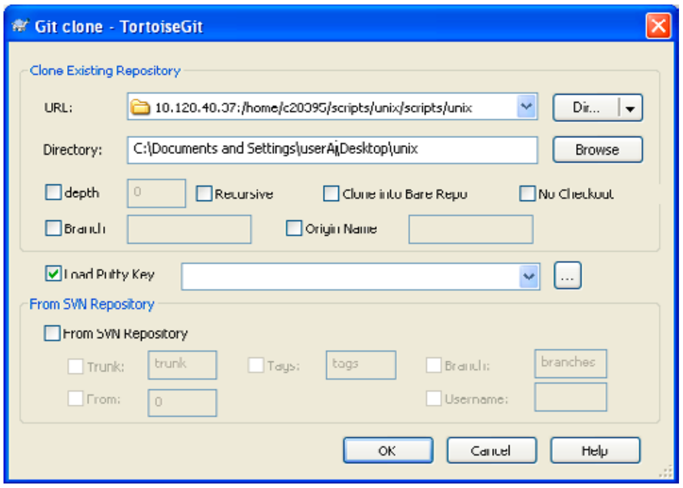
Conclusions

There are some common points shared by CVS, SVN and Git. In fact, both SVN and Git support atomic operations and common versioning, but there are two main differences between the two last ones.



Repository: 10.128.40.37:/home/c20395/scripts/unix/scripts/unix

Figure 13. *Git Browse repository example*



URL: 10.120.40.37:/home/c20395/scripts/unix/scripts/unix
Directory: C:\Documents and Settings\userA\Desktop\unix
[ ] depth 0 [ ] Recursive [ ] Clone into Bare Repo [ ] No Checkout
[ ] Branch [ ] Origin Name
[✓] Load Putty Key
[ ] From SVN Repository
[ ] Trunk: trunk [ ] Tags: tags [ ] Branch: branches
[ ] From: 0 [ ] Username:

Figure 14. *Getting a copy of a remote Git repository*

- Distributed repositories available in a transparent way.
- Use of SSH-based protocols to access remote Git repositories.
- Use of MD5 checksums for version labeling.

In this sense, Git and SVN behave identically as a version refers always to a set of files present during a commit operation. What makes possible the use of Git for high-complexity projects and the syntax and the common procedures for end-users are very similar, so the learning curve is not so steep.

Regardless of whether you are dealing with small projects or large ones, you should consider the use of Git as a powerful replacement for CVS and SVN solutions. If you are looking for a CVS system with excellent reliability against single failures that can damage repositories, Git suits your needs as a software developer perfectly.

References and abbreviations

This section lists the specifications, standards, manuals, and other documents, including policy directives, referenced or used as source material for this plan.

- IEEE Trans. on Soft. Eng. 1975 SE 1-4 – The Source Code Control System (SCCC)
- *http://scm.tigris.org* – Introducción a los sistemas SCM.

Specific Documents

- *http://git-scm.com/book/* – Pro Git Book. Scott Chacon
- *http://git-scm.com/downloads/guis* – Git GUI Clients for MS Windows, Unix and MacOS X
- *http://code.google.com/p/msysgit/downloads/detail?name=Git-1.9.0-preview20140217.exe&can=1&q=%22Full+installer+for+official+Git+for+Windows%22* – Git 1.9.0 tools and GUI for MS Windows
- *https://github.com/gitextensions/gitextensions/* – Git Extensions for MS Windows
- *http://code.google.com/p/tortoisegit/* – TortoiseGit GUI Client for MS Windows

Acronyms and abbreviations

- COTS – Commercial Off-The-Shelf.
- CSCI – Computer Software Configured Item
- CVS – Control Version System
- DAL – Development Assurance Level
- DAV – Distributed Authoring and Versioning
- HTTP – Hypertext Transfer Protocol
- IDE – Integrated Development Environment
- RCS – Revision Control System
- SSL – Secure Sockets Layer
- SVN – Subversion
- VCS – Version Control System

JOSÉ B. ALÓS

José B. Alós has developed an important part of his professional career since 1999 as an EDS employee, as a UNIX System Administrator, mainly focused on SunOS/Solaris, BSD and GNU/Linux. Five years ago he joined EADS Defense and Security, nowadays CASSIDIAN as the person responsible for providing support for end-users in aircraft engineering departments for long-term projects. That is the main reason underneath this article as VAX/VMS systems still play a paramount role in today's aerospace industry for a wide variety of embedded RT systems conceived for mission and flight operations. He was also Assistant Professor in the Universidad de Zaragoza (Spain), specialized in the design of High Availability solutions, and his academic background includes a PhD in Nuclear Engineering and three MsCs in Electrical and Mechanical Engineering, Theoretical Physics and Applied Mathematics.

a d v e r t i s e m e n t

better safe than sorry
www.demyo.com

Working with Hammer File System and PFSes

DragonFly belongs to the same class of operating systems as other BSD-derived systems and Linux. It is based on the same UNIX ideals and APIs and shares ancestor code with other BSD operating systems. DragonFly provides an opportunity for the BSD base to grow in an entirely different direction from the one taken in the FreeBSD, NetBSD, and OpenBSD series.

What you will learn…

- Use dports to install software on DragonFly
- Create Pseudo File Systems in Hammer
- Configure Pseudo File System Snapshots
- Configure other tuning parameters of PFS
- Configure PFS mirroring to a second PFS

What you should know…

- Basic unix shell commands
- Setup SSH key based authentication

DragonFly includes many useful features that differentiate it from other operating systems in the same class. The most prominent one is HAMMER, our modern high performance file system with built-in mirroring and historic access functionality. If you have no dragonfly system installed, please visit *http://www.dragonflybsd.org/download/* and get the latest release and install it on your computer. Most computers now are 64-bit so choose the architecture carefully. Your install system can be a virtual machine on your computer too. The installation is quite straightforward. Remember to choose the default Hammer file system during the install.

Once the installation is complete if you want to SSH into it as `root`, you will have to copy your SSH public key and enter it in the `/root/.ssh/authorized_keys` file of your newly installed system. If you want to copy the key file using SSH, you should make the following changes to `/etc/ssh/sshd_config` and revert once your keys are copied. You may also copy the key using a USB stick without making these changes.

```
PermitRootLogin yes
PasswordAuthentication yes
```

If your system hangs while installing or during boot post-install, then you could try disabling ACPI.

During boot, press 4 in the boot menu to boot ACPI disabled. If it succeeds, make the change permanent by putting these lines in `/boot/loader.conf`

```
#disable acpi
lunset acpi_load
set hint.acpi.0.disabled=1
```

If you now visit `/pfs` you will see the following Pseudo File Systems created by the installer. A Pseudo File System is a Hammer File system inside a Mother hammer which has almost all the functionalities of the mother file system.

```
# cd /pfs && ls
home   tmp   usr   usr.obj   var   var.crash   var.tmp
```

If you visit `/etc/fstab`, you will find where these PFSes are mounted (Listing 1). Now a `df -h` should show you the mount points and PFS mounts (see Listing 2).

You can get the details of all PFSes in the system using the command:

```
# hammer info
```

Before we go in detail to PFSes we will now see how to install third party software in dragonfly. The recent releases use dports unlike the older releases which used pkgsrc from the NetBSD project. In order to install dports on your system, use the following commands:

```
# cd /usr
# make dports-update
# rm /usr/local/etc/pkg.conf
# cd /usr/local/etc/pkg/repos && mv df-latest.conf.sample
   df-latest.conf
```

In order to install third party software, use the following commands:

```
# pkg update
# pkg install "software"
```

Working with Hammer Pseudo File Systems

Let us now create a PFS. By convention PFSes are created under the directory pfs under the mother file system (see Listing 3). Each PFS has a `unique-uuid` and `shared-uuid`. *In order to mirror data from a master PFS, a slave PFS should have the same 'shared-uuid' as the master.*

Now let us look at the different configuration options for a PFS. You can edit the defaults using `hammer viconfig <pfs>` (see Listing 4).

For a new PFS, we find that there is no default configuration but find that there are 6 configuration options commented for the PFS. You will need to uncomment the option in order for it to work. These configuration options will be used by `hammer cleanup` when it is run by periodic daily, periodic weekly and periodic monthly. You can configure the time they run in `/etc/crontab`.

```
# cat /etc/crontab |grep periodic
1     3      *      *      *      root    periodic daily
```

Listing 1. *File System Table*

```
# cat /etc/fstab
# Device                       Mountpoint      FStype     Options      Dump     Pass#
/dev/serno/9QMAS4SZ.s1a          /boot           ufs          rw          1        1
/dev/serno/9QMAS4SZ.s1b          none            swap         sw          0        0
/dev/serno/9QMAS4SZ.s1d          /               hammer       rw          1        1
/pfs/var                       /var              null         rw          0        0
/pfs/tmp                       /tmp              null         rw          0        0
/pfs/usr                       /usr              null         rw          0        0
/pfs/home                      /home             null         rw          0        0
/pfs/usr.obj                   /usr/obj          null         rw          0        0
/pfs/var.crash                 /var/crash        null         rw          0        0
/pfs/var.tmp                   /var/tmp          null         rw          0        0
proc                           /proc             procfs       rw          0        0
```

Listing 2. *File System Mount Points*

```
# df -h
Filesystem                 Size   Used  Avail Capacity  Mounted on
ROOT                       455G    12G   443G     3%    /
devfs                      1.0K   1.0K     0B   100%    /dev
/dev/serno/9QMAS4SZ.s1a    756M   430M   265M    62%    /boot
/pfs/@@-1:00001            455G    12G   443G     3%    /var
/pfs/@@-1:00002            455G    12G   443G     3%    /tmp
/pfs/@@-1:00003            455G    12G   443G     3%    /usr
/pfs/@@-1:00004            455G    12G   443G     3%    /home
/pfs/@@-1:00005            455G    12G   443G     3%    /usr/obj
/pfs/@@-1:00006            455G    12G   443G     3%    /var/crash
/pfs/@@-1:00007            455G    12G   443G     3%    /var/tmp
procfs                     4.0K   4.0K     0B   100%    /proc
```

Listing 3. *Creating a Master PFS*

```
# cd /pfs
# ls
home            tmp             usr             usr.obj         var             var.crash       var.tmp

# hammer pfs-master Data1
Creating PFS #8 succeeded!
Data1
    sync-beg-tid=0x0000000000000001
    sync-end-tid=0x00000002bcc53fb0
    shared-uuid=19345892-d41b-11e3-9552-4587fc6b58f4
    unique-uuid=193458c4-d41b-11e3-9552-4587fc6b58f4
    label=""
    prune-min=00:00:00
    operating as a MASTER
    snapshots directory defaults to /var/hammer/<pfs>
```

Listing 4. *Configuring PFS*

```
# hammer viconfig /pfs/Data1

# No configuration present, here are some defaults
# you can uncomment.  Also remove these instructions
#
#snapshots 1d 60d
#prune     1d 5m
#rebalance 1d 5m
#dedup     1d 5m
#reblock   1d 5m
#recopy    30d 10m
```

Listing 5. *Snapshot information stored in FS Metadata*

```
# cd /pfs/vms4-lxc && ls -l
total 0
drwxr-xr-x  1 root  wheel  0 Nov  3  2012 vms4
drwxr-xr-x  1 root  wheel  0 Nov  3  2012 vmsx4
# date
Mon May 12 12:34:32 IST 2014
# hammer snap /pfs/vms4-lxc
# date
Mon May 12 12:34:57 IST 2014
# hammer snapls /pfs/vms4-lxc |grep "12:34"
0x0000000b98415a50      2014-05-12 12:34:53 IST -
# cd /pfs/vms4-lxc/@@0x0000000b98415a50
# ls -l
total 0
drwxr-xr-x  1 root  wheel  0 Nov  5  2012 vms4
drwxr-xr-x  1 root  wheel  0 Nov  5  2012 vmsx4
```

Listing 6. *Pruning a PFS*

```
# hammer prune /pfs/vms4-lxc
TID 0000000b98415a50 - 0000000b9841e0a0
TID 0000000b983c2cd0 - 0000000b98415a50
TID 0000000b983a2390 - 0000000b983c2cd0
TID 0000000b983a2170 - 0000000b983a2390
TID 0000000b9839a080 - 0000000b983a2170
TID 0000000b980daa40 - 0000000b9839a080
TID 0000000b97d437a0 - 0000000b980daa40
TID 0000000b979c3640 - 0000000b97d437a0
TID 0000000000000001 - 0000000b979c3640
Prune //: 9 snapshots
Prune //: objspace 8000000000000000:0000
   7fffffffffffffff:ffff pfs_id 0
Prune //: prune_min is 0d/00:00:00
Prune // succeeded
Pruned 0/6387 records (0 directory entries) and 0 bytes
```

Listing 7. *Reblancing B-Tree*

```
# hammer rebalance /pfs/vms4-lxc
rebalance start 8000000000000000:0000
Rebalance /pfs/vms4-lxc succeeded
Rebalance:
    370044 btree nodes scanned
    131 btree nodes deleted
    0 collision retries
    61349 btree nodes rebalanced

# hammer dedup <pfs>
```

```
15    4     *     *     6     root    periodic weekly
30    5     1     *     *     root    periodic monthly
```

Let us look at the first option `snapshots`.

The option `1d` tells `hammer cleanup` that a snapshot of the PFS should be taken every 24 hours. The option `60d` tells 'hammer cleanup' to retain snapshots until they are 60 days old. Snapshots older than 60 days will be removed in this configuration. The second option in `prune`.

By default the Hammer file system retains every change made to it. The prune option tells 'hammer cleanup' to make the snapshots "fine grained" (i.e., all changes that happened between the snapshots will be removed).

The option `1d` tells 'hammer cleanup' that the PFS snapshots should be pruned every 24 hours. The option `5m` tells `hammer cleanup` that the pruning should only happen for 5 minutes. The third option is `rebalance`. It tells 'hammer cleanup' to rebalance the B-Tree. Nodes with a small number of elements will be combined and element counts will be smoothed out between nodes. The option `1d` tells hammer cleanup to rebalance the B-Tree every `24` hours. The option `5m` tells hammer cleanup to run rebalance only for 5 minutes.

The fourth option is `dedup`. This tells `hammer clean` up to deduplicate the PFS. Dedup runs on block level and removes duplicate data from the PFS. This option is very useful if your PFS contains many similar copies of files. The option `1d` tells the hammer cleanup to run deduplication on the PFS every 24 hours. The option `5m` limits the deduplication to 5 minutes. The fifth option is `reblock`**.**

This option tells 'hammer cleanup' to defragment the PFS to a level of *95%* and free up space for reuse. The option `1d` tells `hammer cleanup` to reblock every 24 hours. The option `5m` tells hammer cleanup to run reblock for 5 minutes. The sixth option is `recopy.`

This option tells hammer cleanup to do a *full defragmentation* of the PFS. The option `30d` tells 'hammer cleanup' to run recopy every month. The option `10m` tells 'hammer cleanup' to run recopy for 10 minutes.

These parameters must be retuned according to the data, the volume of data and the frequency of writes inside the PFS. For example one of my PFSes with massive Data has the following configuration:

```
# hammer config /pfs/Data
snapshots 1d 30d
prune     1d 15m
rebalance 1d 5m
dedup     1d 25m
reblock   1d 15m
recopy    30d 15m
```

These can be run manually as the following commands:

```
# hammer snap <pfs>
```

Let us manually snapshot a PFS `vms4-lxc`. If no snapshot directory is mentioned, information on how to access the snapshot will be stored in the filesystem metadata (see Listing 5). Here we see that the command `hammer snap <pfs>` is used to snapshot a PFS. The command `hammer snapls <pfs>` is used to view the snapshots. And the snapshot ID with `@@` appended to its front can be used to browse the snapshot contents.

In order to have a symlink to browse the snapshots, one can use the format:

```
# hammer snapshot <pfs> <alternate snapshot symlink
   directory> "tag"
```

The `hammer cleanup` utility, on the other hand, automatically creates a SymLink in `/var/hammer/<pfs>` when it is run through periodic

```
# hammer prune <pfs>
```

Let us now see how manual pruning works on a PFS (Listing 6). Prune removes all the changes in the file systems between snapshots and also the snapshots that have expired. It also shows the pruning details.

```
# hammer rebalance <pfs>
```

Now let us rebalance the B-Tree manually (see Listing 7).

Before running hammer dedup manually it is possible to find how much space will be freed using the command

```
# hammer dedup-simulate
```

```
# hammer dedup-simulate /pfs/vms4-lxc
Dedup-simulate running
Dedup-simulate /pfs/vms4-lxc succeeded
Simulated dedup ratio = 2.45
```

The higher the dedup ratio, the larger the amount of Data that can be contained in much less space.

More real world scenarios where `hammer dedup` was helpful can be found in the following two links:

- *http://lists.dragonflybsd.org/pipermail/users/2011-July/087725.html*
- *http://leaf.dragonflybsd.org/mailarchive/users/2011-07/msg00026.html* (Listing 8)

After the de-dup we find that only 87 GB were used to contain 214 GB of Data.

```
# hammer reblock <pfs> <fill% >
```

Let us now defragment the PFS (Listing 9). When manually run, the default fill percentage is 100% and will cause the file system to be completely defragmented. All specified element types will be reallocated and rewritten. If you wish to quickly free up space instead you can do so by specifying a smaller fill percentage, such as 90% or 80% (the `%` suffix is not needed).

How To Implement Hammer Pseudo File System(PFS) Slave Mirroring From PFS Master

I have two 500 GB hard disks both with the Hammer file system. I want to create a master PFS on one hard disk and a slave PFS on the other disk. I want to mirror the data continuously from the master PFS to the slave PFS. This will help me avoid long 'fsck' and RAID parity rewrite times after an unclean shutdown, and also will give me a setup somewhat like RAID 1.

Creating the master PFS on Disk 1

The Hammer file systems on Disk 1 and Disk 2 are mounted in `/etc/fstab` according to the following.

```
/dev/ad4s1h         /Backup1       hammer   rw     2        2
/dev/ad6s1h         /Backup2       hammer   rw     2        2
```

Go to the Hammer file system on Disk 1. We will be creating a master PFS called `test` and will be mounting it using a null mount. If you don't have a directory called `pfs` under the Hammer file system you should create it.

```
# pwd
/Backup1
# mkdir pfs
```

If you already have the pfs directory under the Hammer file system you can skip the above step and continue (see Listing 10). Now the master PFS 'test' is created. Make a note of its 'shared-uuid' because we will need to use that to create the slave PFS for mirroring. You can mount the PFS under the Hammer file system on Disk 1 by doing the following:

Listing 8. *De-duplicating a PFS*

```
# hammer dedup /pfs/vms4-lxc
Dedup running
Dedup /pfs/vms4-lxc succeeded
Dedup ratio = 2.45
      214 GB referenced
       87 GB allocated
       18 MB skipped
        597 CRC collisions
          0 SHA collisions
          3 bigblock underflows
          0 new dedup records
          0 new dedup bytes
```

Listing 9. *Defragmenting PFS*

```
# hammer reblock /pfs/vms4-lxc/
reblock start 8000000000000000:0000 free level 0

Reblock /pfs/vms4-lxc/ succeeded
Reblocked:
    187016/187017 btree nodes
    4404440/9444810 data elements
    47942945645/231120183341 data bytes
```

Listing 10. *Creating a Master PFS to mirror from*

```
# hammer pfs-master /Backup1/pfs/test
Creating PFS #3 succeeded!
/Backup1/pfs/test
sync-beg-tid=0x0000000000000001
sync-end-tid=0x000000013f644ce0
shared-uuid=9043570e-b3d9-11de-9bef-011617202aa6
unique-uuid=9043574c-b3d9-11de-9bef-011617202aa6
label=""
prune-min=00:00:00
operating as a MASTER
snapshots dir for master defaults to <fs>/snapshots
```

Listing 11. *Creating a Slave PFS to mirror from Master*

```
# hammer pfs-slave /Backup2/pfs/test shared-
   uuid=9043570e-b3d9-11de-9bef-011617202aa6
Creating PFS #3 succeeded!
/Backup2/pfs/test
sync-beg-tid=0x0000000000000001
sync-end-tid=0x0000000000000001
shared-uuid=9043570e-b3d9-11de-9bef-011617202aa6
unique-uuid=97d77f53-b3da-11de-9bef-011617202aa6
slave
label=""
prune-min=00:00:00
operating as a SLAVE
snapshots directory not set for slave
```

```
# mkdir /Backup1/test
```

Now Edit `/etc/fstab` to contain the following line.

```
/Backup1/pfs/test   /Backup1/test    null    rw      0   0
```

Now mount the PFS and verify by doing.

```
# mount -a
# mount |grep test
/Backup1/pfs/@@-1:00003 on /Backup1/test (null, local)
```

Creating the slave PFS on Disk 2

Note that we must use the `shared-uuid` of the master PFS to enable mirroring (see Listing 11). The slave PFS is not mounted but a symlink can be created in the root Hammer file system to point to it.

```
# ln -s /Backup2/pfs/test /Backup2/test
# ls -l /Backup2/test
lrwxr-xr-x  1 root  wheel  17 Oct  8 12:07 /Backup2/test
   -> /Backup2/pfs/test
```

(This step is optional, the PFS can be read through the original magic symlink `/Backup2/pfs/test`).

Copying contents from PFS on Disk 1 to PFS on Disk 2 to enable mirroring

The slave PFS will be accessible only after the first `mirror-copy` operation.

```
# touch /Backup1/test/test-file
# ls /Backup1/test/
test-file
# sync
```

We do the "sync" so that the file creation operation is flushed from the kernel memory. Mirroring works only on operations flushed from the kernel memory. The slave PFS will be accessible only after the first mirroring operation.

```
# hammer mirror-copy /Backup1/test /Backup2/pfs/test
histogram range 000000013f6425fd - 000000013f644d60
Mirror-read: Mirror from 0000000000000002 to
   000000013f644d60
Mirror-read /Backup1/test succeeded
# ls /Backup2/test/
test-file
```

Enabling continuous mirroring. The `hammer mirror-stream` will automatically restart if the connection is lost so you only need to start it up once at boot. You can do this with an `@reboot` entry in the `crontab`.

```
@reboot /sbin/hammer mirror-stream /Backup1/test
   /Backup2/test
```

If the disk on which the slave PFS resides goes offline then another disk can be attached and another slave PFS can be configured as said above. If the disk on which the master PFS resides goes down, then the slave PFS can be upgraded to become a master PFS using the command:

```
# hammer pfs-upgrade <slave pfs>
```

and writes continued to it. Another disk can be attached and another slave PFS can be configured to this new master PFS.

PFS Mirroring through a network

The slave PFS need not be on the same system as the master PFS. The system where the slave PFS resides can be many miles apart, even in another continent, if it is connected to the system in which the master PFS resides through a network. If the IP address of a system with master PFS is `10.0.1.1` then an entry as shown below in the crontab of the system in which the slave PFS resides will start and continue remote mirroring when the system with the slave PFS boots up.

```
@reboot /sbin/hammer mirror-stream root@10.0.1.1:/Backup1/
   test /Backup2/test
```

For this to work SSH key based access should be enabled from the slave PFS system's `root` user to the master PFS system's `root` user by copying the contents of the slave PFS system's `/root/.ssh/id_rsa.pub` file to the master PFS system's `/root/.ssh/authorized_keys` file.

Finally in order to `delete` a PFS simply `rm -rf` won't do. You have to use the command:

```
# hammer pfs-destroy <pfs>
```

Conclusion

The HAMMER file system is a neat implementation of a copy-on file system with extra features. It uses much less RAM than ZFS and it is a good choice for any kind of operation.

SIJU GEORGE

Running Javascript Client and Server Side

with MeteorJS on FreeBSD 11

Javascript is everywhere, in web browsers, servers (NodeJS), smartphones (Phonegap/Cordova). MeteorJS takes advantage of this situation to revolutionize web frameworks. In this article, we'll introduce MeteorJS and demonstrate an example app deployed on FreeBSD 11.

What you will learn…

- Some key features of MeteorJS
- How to build a realtime chat with a few lines of code

What you should know…

- Basic web programming

Meteor (*http://meteor.com*, hashtag *#meteorjs* on Twitter) is an Open Source isomorphic javascript framework. It means that mostly data structures are available on the client and on the server, without the need to plan an API (database everywhere).

This is possible because the same programming language is used on the client (browser) and server and because the framework uses a synchronization protocol (DDP) to fill on-demand the data structures on the client.

Another big benefit is that you don't have duplicated code, because you can share it. Imagine validation functions, url router, HTML template generation... you can decide to execute the code of the client, server or both.

If you want to over-simplify, "Meteor is to NodeJS what Ruby on Rails is to Ruby". It isn't completely true because Meteor isn't only server side like other NodeJS frameworks, but it has the philosophy of Convention Over Configuration, so you can immediately concentrate on solving your domain problem and eventually override that behavior later.

Other important pieces are MongoDB/Minimongo which allow the management of the database in javascript and the Future library which reduce the "callback hell" javascript effect.

Installation on GNU/Linux or Mac OS X (Development mode)

Meteor runs on known multiplatform UNIX/Linux Open Source software, but currently only GNU/Linux and Mac OS X are the official supported platforms for development. This isn't a big problem when you deploy a Meteor app on FreeBSD server, because it became almost a NodeJS + MongoDB app, but we'll see it later.

So type in the terminal:

```
$ curl https://install.meteor.com/ | sh
```

Running the command above on a supported platform will install a bundled Meteor with all the dependencies needed.

The "Minimal Chat" example

The Realtime applications, like chat, is really where Meteor shines in its simplicity. All the workflow will be around the terminal (with meteor command running), the web browser and your preferred text-editor.

Let's create a new app:

```
$ meteor create ChatApp
$ cd ChatApp
```

Listing 1. *ChatApp.html*

```
<!-- ChatApp.html -->
<head>
  <title>Meteor Minimal Chat</title>
</head>

<body>
  {{> chatList}}
  {{> chatEntry}}
</body>

<template name="chatList">
  <h1>Messages</h1>

  {{#each messages}}
    <p>
      <strong>{{user}}:</strong> {{text}}
    </p>
  {{/each}}
</template>

<template name="chatEntry">
  <input  id="user" placeholder="username" />
  <input  id="text" placeholder="your text" />
  <button id="send">Send</button>
</template>
```

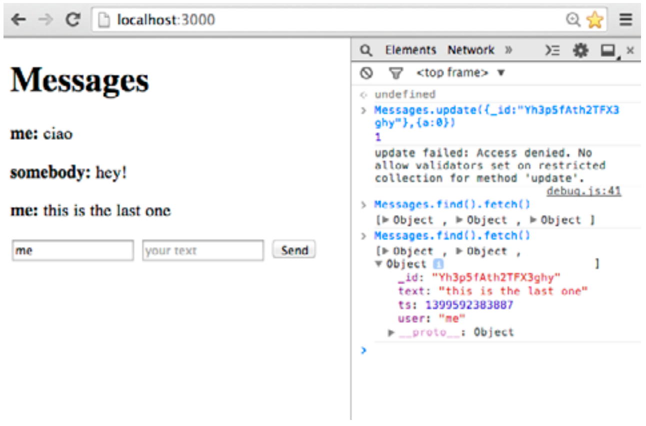
Listing 2. *ChatApp.js*

```
// ChatApp.js
Messages = new Meteor.Collection('messages');

if (Meteor.isClient){
  Template.chatList.helpers({
    'messages': function () {
      return Messages.find({});
    }
  });

  Template.chatEntry.events({
    'click #send' : function () {
      Messages.insert(
        { user:$('#user').val(), text:$('#text').val() }
      );
      $('#text').val('');
    }
  });
}                   4.0K   4.0K     0B   100%    /proc
```

Then replace the files like this: Listing 1 and Listing 2. Now run the meteor command again without arguments:

```
~/ChatApp ★  meteor

[[[[[ ~/ChatApp ]]]]]

=> Started proxy.
=> Started MongoDB
=> Started your app.

=> App running at: http://localhost:3000/
```

If you open your web browser at *http://localhost:3000/*, you should see a working chat, but let's explain what happened.



Messages

me: ciao

somebody: hey!

me: this is the last one

```
undefined
> Messages.update({_id:"Yh3p5fAth2TFX3
  ghy"},{a:0})
  1
  update failed: Access denied. No
  allow validators set on restricted
  collection for method 'update'.
                                debug.js:41
> Messages.find().fetch()
  [▶ Object , ▶ Object , ▶ Object ]
> Messages.find().fetch()
  [▶ Object , ▶ Object ,
  ▼ Object                      ]
      _id: "Yh3p5fAth2TFX3ghy"
      text: "this is the last one"
      ts: 1399592383887
      user: "me"
    ▶ __proto__: Object
>
```

Figure 1. *The running app with a javascript console*

ChatApp.html looks like an HTML file but it isn't. It is HTML plus Spacebars (a Handlebars / Mustache -like syntax). *chatList* and *chatEntry* are the partial templates. Then it isn't used directly but is compiled to a javascript function. You can see it by opening the web browser console (command + shift + c on Chrome) and typing `Template.chatEntry()`.

ChatApp.js is a javascript file where *Messages* is defined, the collections where the data are stored in a persistent way on the server, and where the user events are bound to the templates. *messages* is the cursor iterator used by the `{{#each messages}}` cycle in the template and every time new data is added or changed, the template is partially or completely re-rendered. By default Meteor is also magic, so let's remove some packages to better understand what happens.

```
meteor remove autopublish insecure
```

- *autopublish* is responsible for syncing/filling the collection on the client. In the real world, the client should only see the documents (records for SQL users) that it really needs for performance and security reasons.
- *insecure* is the package which doesn't restrict the actions that a user could perform on the data (insert, update, delete).

If we try again now, the application doesn't show any message and if we try to insert a new one, we get in the web browser console: *insert failed: Access denied*. So, let's evolve the chat application. We create the *client/* and *server/* directories so we don't need to check where we are with *Meteor.isClient* and the code on the *server/* will never be sent to the client.

```
$ mkdir {client,server}
```

Now the messages are back and we can also decide that the client will see just the latest 10 messages. Just replace the cursor with *return* `Messages.find({}, {count:10})`. We still can't insert new messages; let's fix it by adding an explicit rule that anybody can insert new messages. Reopen the *ChatApp.js* file and add these new lines below the Messages definition: Listing 4.

Meteor has core packages to handle authentication with password or OAuth2 providers (like Facebook or Twitter) but we won't use them in this example, this is why *userId*

Listing 3. *Client/Subscribe.js*

```
// client/subscribe.js
Meteor.subscribe('allMessages');

// server/publish.js
Meteor.publish('allMessages', function (opts) {
  return Messages.find({});
});
```

Listing 4. *ChatApp.js*

```
// ChatApp.js
Messages = new Meteor.Collection('messages');
Messages.allow({
  insert: function (userId, doc) {
    doc.ts = (new Date()).getTime();
    return true;
  }
});

[...]
```

is present. *doc* is the new document that will be added to the collection. So we could analyze, add other fields or validate it before the insertion in the database.

In the example, a timestamp (*ts*) is added to every new message, so changing the publication (in *publish.js*) you can publish just the last 3 messages on the client.

To apply these changes quickly and correctly, you need to remove the old messages without the timestamp with "meteor reset".

```
// server/publish.js
Meteor.publish('allMessages', function (opts) {
  return Messages.find({}, {sort:{ts:-1}, limit:3 });
});
```

You can check that the other messages really aren't available on the client by checking the javascript console again:

```
Messages.find().fetch()
->[Object, Object, Object]
```

This is how Meteor manages the data, even if the data structures respond to the same methods the effective data available on the client depends on how the *publish* and *subscribe* methods are managed...and typing

```
$ meteor deploy minichat.meteor.com
```

We obtain a live, public application that is available worldwide. You can really try this demo app at *http://minichat.meteor.com.*

Deploy on FreeBSD

As we said, the Meteor application can be considered a special NodeJS application. So if we want to deploy the app in our server instead, another command is useful that will pack everything needed.

```
$ meteor deploy chat.tgz
```

Now we can move and unpack *chat.tgz* to the FreeBSD server and start with the system set up.

Notes: The latest MongoDB production release is 2.6.x. Currently Meteor 0.8.x and FreeBSD 11 support MongoDB 2.4.x. I tried these steps on FreeBSD 11 but they may also work on FreeBSD 10.

I assume you can use *sudo* or you are root on the system.

Installation of Meteor dependencies

```
$ sudo pkg install -y git node npm mongodb python gmake
```

A Node dependency needs to be recompiled on FreeBSD with g++ but clang++ (which is already installed) works in this situation. If you really need g++ install it, otherwise:

```
$ sudo ln -s /usr/bin/clang++ /usr/bin/g++
```

Now we need to start the database

```
$ sudo service mongodb onestart
```

It will start the MongoDB daemon, in a non-persistent way, if you want it at the next reboot add `mongod_enable="YES"` to `/etc/rc.conf`.

The ground is ready, now you need a Meteor app to run. Where you unpacked *chat.tgz*, there should be a *bundle/* directory.

Run a Meteor app from a bundle

This bundle also contains some Linux binaries, which must be rebuilt for FreeBSD.

```
$ cd bundle/server/node_modules # or bundle/programs/
   server/node_modules/ on older Meteor versions
$ rm -rf fibers
$ npm install fibers@1.0.1
$ cd -
```

Now you should be ready to run a basic Meteor app on FreeBSD; look at the official documentation for further env variables and tweaks.

```
$ PORT=3000 MONGO_URL="mongodb://localhost:27017/myappdb"
   node bundle/main.js
```

Resources

- *http://docs.meteor.com* – the official Meteor documentation
- *https://atmospherejs.com* – The community packages to extend Meteor
- *http://madewith.meteor.com* – A gallery of apps and experiments built with Meteor
- *http://discovermeteor.com* – A Meteor book continuosly updated
- *http://weuse.meteor.com* – Where are the Meteor users on a map
- *http://meteorhacks.com* – A blog about Meteor and its community

Finally, you should see your Meteor app at *http://your-FreeBSD-ip:3000.*

Summary

We've just scratched the surface of the MeteorJS features. The realtime web application or single page application (SPA) are quite different from the canonical PHP web pages; but as we've seen, Meteor introduces new patterns that will save you many lines of code.

The chat app we built will also provide the messages to the clients connected and will keep the data continuously updated without the need of explicit AJAX calls or the need to handle the reconnections by ourselves.

LUIGI MASELLI

Luigi Maselli (hackerpreneur, freelance developer, passionate about the web and UNIX/Linux, blogger at http://grigio.org).

a d v e r t i s e m e n t

IT-Securityguard

Lets secure IT

Android Vulnerability Scan

Web Penetration testing

Secure hosting

Understanding bhyve, the BSD hypervisor

For a long time now, virtualization has been an important part of an information system. FreeBSD has "jails", and in a lot of cases, it's ideal, secure and flexible. But, sometimes we need to use other operating systems. We need bhyve, the "BSD hypervisor", which is a legacy-free hypervisor/virtual machine manager developed on FreeBSD.

What you will learn…

- basics about freebsd virtualization
- introduction to bhyve and it state
- discover works and works in progress about it
- create your first virtual machine

What you should know…

- basic unix shell commands
- basic understand of virtualization

A hypervisor allows the operation of one or more guest operating systems within a host operating system. bhyve was merged from the bhyve projects branch into the FreeBSD head on January 19th, 2013. It was officially released on January 20th, 2014 as part of FreeBSD 10.0. It is mainly developed by Peter Grehan [12] and Neel Natu.

In this article, we will try to understand what bhyve is and we will prepare "Playing with bhyve, the BSD hypervisor", the second part of this article where we will create guests from different OSes in a FreeBSD host with bhyve.

General Considerations

bhyve is a Type-2 hypervisor (Type 1 native/bare metal hypervisor (VMware, Virtualbox …) Type 2 hosted hypervisor (Xen, KVM …) that runs on the FreeBSD platform. It currently only runs FreeBSD (9.x or later) and Linux guests. Current development efforts aim at widening support for other x86 64-bit operating systems. After a great deal of work by all involved parties, bhyve was shipped as part of FreeBSD 10.0-RELEASE. Increased interest in bhyve and the first usable version have provided great feedback and many bug reports. Sometimes, you will still see the terms "BhyVe", but the developers hopefully depreciated the name "BHyVe" and simply refer to it as "bhyve" (hopefully :p). And don't miss the FreeBSD handbook part about bhyve [26].

Hardware Considerations

bhyve currently supports Intel processors with Extended Page Tables (most of Atom C2000, Core i3, i5, i7 and related Xeon processors are supported). AMD RVI (Rapid Virtualization Indexing, formerly Nested Page Tables) support currently resides in a separate development branch. "Barcelona" class and newer AMD processors should include the required RVI extension and as with Intel processors, the presence of the "POPCNT" (POP Count) processor feature in dmesg (8) will also indicate RVI support.

For example in the dmesg of a FreeBSD 10.0-RELEASE-p2, you will see for an i7: Listing 1.

Nested paging support is available in bhyve starting from FreeBSD 10.0 and provides useful features such as transparent superpages and overprovisioning of guest memory. Nested paging is a hardware technique used to reduce the overhead of memory virtualization. Specifically, this refers to Intel EPT (Extended Page Tables) and AMD NPT (Nested Page Tables). During AsiaBSDCon 2014 [15], Neel Natu and Peter Grehan gave an excellent presentation called « Nested Paging in bhyve » [10]. You can run bhyve under VMware nested VT-x but you must enable it. bhyve support of nested VT-x (bhyve under bhyve) is investigated. PCI passthru with bhyve works on systems that have the Intel IOMMU (aka VT-d feature/support). This can be determined by looking for a DMAR entry in the ACPI tables (ex: acpidump -t | grep DMAR).

bhyve supports AHCI devices, which improve performance thanks to non-blocking I/O. Raw disk images and any block device such as ZFS zvols and iSCSI targets, plus ISOs using the AHCI driver are supported as boot media.

Because it would simplify the booting of non-FreeBSD OSes and integrated video support implementation, bhyve would support UEFI/BIOS (BSD-licensed BIOS replacement) [23]. And closely related to UEFI/BIOS support, bhyve will support VGA graphics.

bhyve has an "out-of-band"/"lights-out management" serial console, which can be accessed in several ways with your preferred nullmodem/pty. Stdio output can also be sent to a terminal multiplexer like tmux [24] or screen [25].

bhyve has ACPI support; proper shutdown of guest OSes has been implemented with SVN revision 259826. ACPI suspend and resume features are currently under development [21]. As a legacy free hypervisor, a bhyve host requires the Extended Page Tables (EPT) feature found on "Nehalem" and newer generations of Intel processors. This requirement eliminates the need for memory management routines that are traditionally implemented in software and yields virtually bare metal guest performance. A bhyve guest requires VirtIO network and block devices drivers, which were already available in FreeBSD 8/9/10 at the time of bhyve's import.

Software Considerations

An overview of how it works from the bhyve web site [30]: see Figure 1.

bhyve can be built and operated on FreeBSD 9.0 amd64 but its active development is taking place on 11-CURRENT amd64. bhyve is included in FreeBSD 10.0-RELEASE amd64. Since the commit r258141 on November 14th, 2013, FreeBSD has its own allocation of IEEE Organizationally Unique Identifiers (check «Search the Public MA-L Listing» from the IEEE [29]) and bhyve will be allocated a sub-set of these and have its own MAC address allocation.

```
58-9C-FC (hex)        FreeBSD Foundation
589CFC   (base 16)    FreeBSD Foundation
                         P.O. Box 20247
                         Boulder CO  80308-3247
                         UNITED STATES
```

bhyve uses the VirtIO (MSI and MSI-X) set of devices for virtualized storage and network devices. bhyve supports MSI and MSI-X interrupts although MSI interrupts are not part of the VirtIO specification.

bhyve supports any version of FreeBSD amd64 with VirtIO support (kern.vm_guest sysctl will report if FreeBSD is under KVM or bhyve.), plus OpenBSD amd64, and GNU/Linux amd64 (some versions of centos / redhat / debian / opensuse / ubuntu have been tested successfully). At the bhyvecon 2014 [14], you have a great example with CentOS in "Depenguinating your infrastructure" [16]. For the story, Illumos support is under active development and bhyve has support for i386 VMs.

bhyve consists of the vmm.ko loadable kernel module (kernel module for VT-x, VT-d and hypervisor control), the libvmmapi (userland API, the front-end to the vmm.ko chardev interface) library, bhyve (userland part of hypervisor, emulates devices), bhyveload (loads guest operating system) and bhyvectl (a management tool) utilities (in total these binaries are under 500K). bhyveload only supports FreeBSD. You need another OS Loader to support

Listing 1. *dmesg extract*

```
CPU: Intel(R) Core(TM) i7 CPU         950  @ 3.07GHz (3103.64-MHz K8-class CPU)
  Origin = "GenuineIntel"  Id = 0x106a5  Family = 0x6  Model = 0x1a  Stepping = 5
Features=0xbfebfbff<FPU,VME,DE,PSE,TSC,MSR,PAE,MCE,CX8,APIC,SEP,MTRR,PGE,MCA,CMOV,PAT,PSE36,CLFLUSH,DTS,ACPI,MMX,FXS
   R,SSE,SSE2,SS,HTT,TM,PBE>
  Features2=0x98e3bd<SSE3,DTES64,MON,DS_CPL,VMX,EST,TM2,SSSE3,CX16,xTPR,PDCM,SSE4.1,SSE4.2,POPCNT>
  AMD Features=0x28100800<SYSCALL,NX,RDTSCP,LM>
  AMD Features2=0x1<LAHF>
  TSC: P-state invariant, performance statistics
```

other OSes. grub2-bhyve [2] is the solution. It's a modified version of grub2, runs on the host OS (FreeBSD) and can load Linux and OpenBSD.

Libvirt [7] includes bhyve support. Enabling bhyve support allows consumers to use bhyve in libvirt-ready applications without major efforts.Currently, libvirt supports almost all essential features of bhyve, such as Virtual Machine lifecycle (start, stop), bridged networking, and virtio/SATA driver support. The work continues to implement more API calls and to cover more features offered by bhyve.

Virt-manager with this commit [5] shows that it supports bhyve too.

Other Virtualisation Stuff Under FreeBSD

Because I can't speak about bhyve without a word about jails, here are just a few words to show some differences.

Jails are isolated virtual instances of FreeBSD that all run over the same kernel. It's a kernel-level virtualization environment. You cannot install another operating system in a jail. Process in jails is just a normal process for the kernel. Isolation between environments is done by some «kernel tricks». Jails share one kernel with all the others – if the kernel panics, all jails are dead. bhyve is a hypervisor, it virtualizes a machine. You can run a full operating system in a virtual machine. From the guest OS, it looks like real hardware. A virtual machine is a normal process for host operating systems. A virtual machine does not share the kernel, it is completely isolated from the others.

bhyve in the «real world»

People do a lot of stuff with or for bhyve. It's time to introduce some examples. To start, bhyve.org provides the script 'vmrun.sh' [1] written by the bhyve developer Neel Natu, and a FreeBSD 10 installation iso image (release.iso); a bootable disk image (diskdev) can be automatically created by the standard FreeBSD installer. To continue, they provide several scripts to make the creation and operation of custom bhyve guests easier.

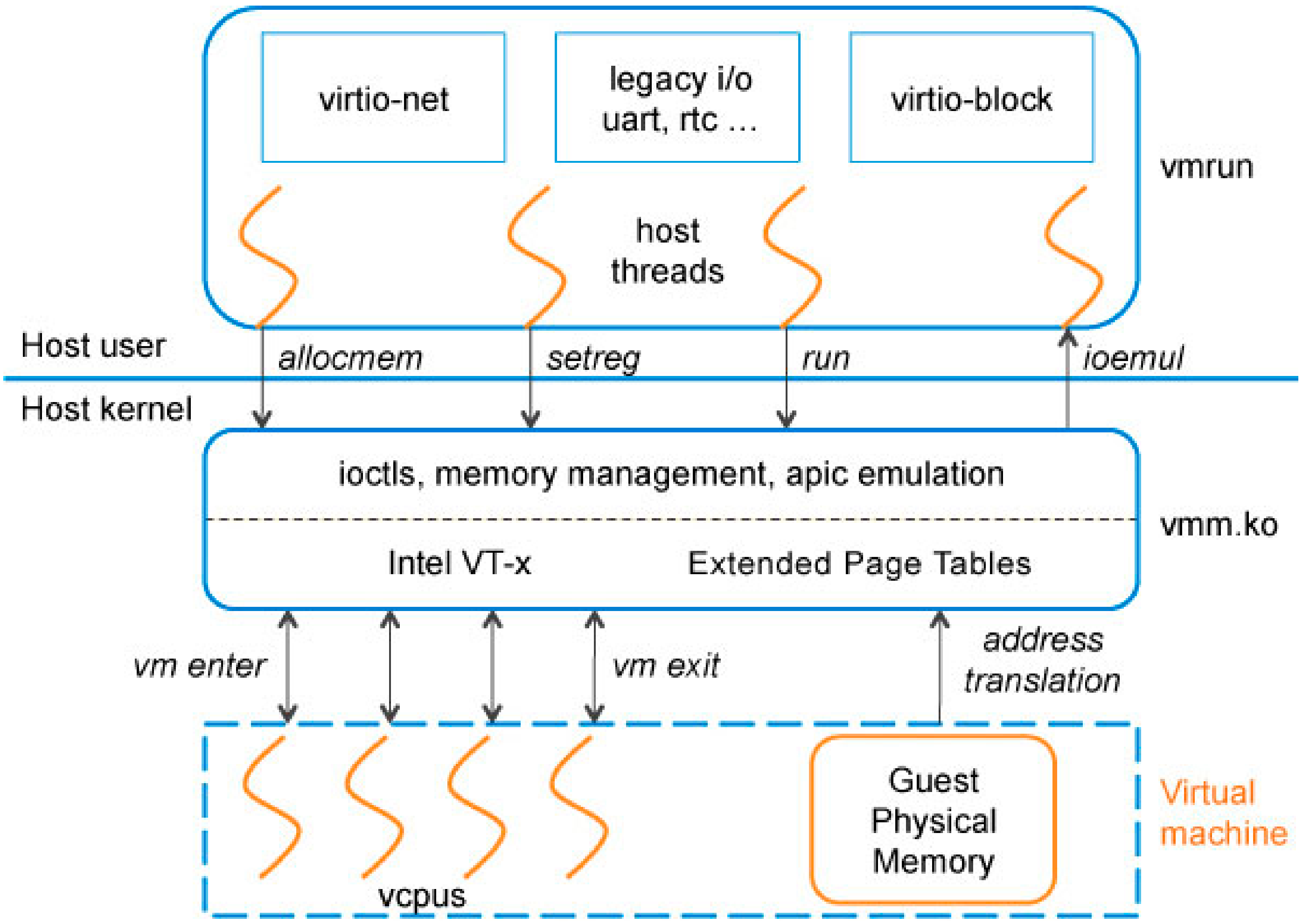


Figure 1. *bhyve implementation*

BSDRouterProject [3] provides some scripts for setting-up labs with. Ezhyve's [6] goal is easier creation of bhyve VMs. CBSD (management layer written for the FreeBSD jails subsystem), since version 10.0.6, has started to work with bhyve.

You can see in the PetiteCloud hypervisor support matrix [4] that bhyve is an option and you can do some set-up with it. PetiteCloud is a 100% Free Open Source and Open Knowledge Cloud Foundation Layer for small non-data center mini-/private-clouds.

From OpenFest 2013, to have some stuff about bhyve but more globally about virtualization, read «the BSD hypervisor» by Paul Schenkeveld [28].

At bhyvecon 2014, I was particularly interested in the concept behind OSv and the couple OSv/bhyve. OSv [17] is a new open-source operating system for virtual-machines designed to execute a single application on top of a hypervisor, resulting in superior performance and effortless management. At the bhyvecon Takuya ASADA (@syuu1228) in «OSv on bhyve» [18], showed us that it can be a working solution.

On March 13, 2014, Craig Rodrigues gave a talk to a packed room during the monthly BAFUG meeting at Hacker Dojo. The talk was about the progress of the effort to date; they have set up the Jenkins Continuous Integration system inside the FreeBSD cluster with the use of bhyve VMs [19].

At AsiaBSDCon 2014, you can read an excellent paper about visualizing Unix system activity using collectd, its site-specific alternatives, Graphite, DTrace and FreeBSD called «Visualizing Unix: Graphing bhyve, ZFS and PF with Graphite» [20] by Michael Dexter.

Create your first virtual machine in bhyve

Let's start by a first minimal VM

```
root@akatsuki:~ # uname -a
FreeBSD akatsuki 10.0-RELEASE-p2 FreeBSD 10.0-RELEASE-p2
    #0: Tue Apr 29 17:06:01 UTC 2014     root@amd64-builder.
    daemonology.net:/usr/obj/usr/src/sys/GENERIC  amd64
root@akatsuki:~ # mkdir /home/vm
root@akatsuki:~ # cd /home/vm
```

load the bhyve kernel module and verify for if_tap presence

```
root@akatsuki:/home/vm # kldload vmm
root@akatsuki:/home/vm # kldstat | grep if_tap
 7    1 0xffffffff81bdf000 5545     if_tap.ko
root@akatsuki:/home/vm #
root@akatsuki:/home/vm # kldstat |grep vmm
 4    1 0xffffffff81a60000 16efe5   vmm.ko
```

My ethernet card is re1. Create a bridge and a tap network interface that you add to the bridge:

```
root@akatsuki:/home/vm # ifconfig bridge0 create
root@akatsuki:/home/vm # ifconfig tap0 create
root@akatsuki:/home/vm # sysctl net.link.tap.up_on_open=1
net.link.tap.up_on_open: 0 -> 1
root@akatsuki:/home/vm # ifconfig bridge0 addm re1 addm tap0
root@akatsuki:/home/vm # ifconfig bridge0 up
root@akatsuki:/home/vm #
```

We download an iso to install:

```
root@akatsuki:/home/vm # fetch ftp://ftp.freebsd.org/pub/
   FreeBSD/ISO-IMAGES-amd64/10.0/FreeBSD-10.0-RELEASE-
   amd64-bootonly.iso
FreeBSD-10.0-RELEASE-amd64-bootonly.iso       100% of  209
   MB 1200 kBps 02m59s
```

We are on FreeBSD 10, let's create a disk image to use for the VM:

```
root@akatsuki:/home/vm # truncate -s 25g bsdmag.img
root@akatsuki:/home/vm # ls -la
total 104
drwxr-xr-x  2 root  wheel          512 May  6 23:12 .
drwxr-xr-x  3 root  wheel          512 May  6 23:10 ..
-rw-r--r--  1 root  wheel  26843545600 May  6 23:12
   bsdmag.img
```

We copy/modify the script written by bhyve developer neel@:

```
root@akatsuki:/home/vm # cp /usr/share/examples/bhyve/
   vmrun.sh bhyvebsdmag.sh
root@akatsuki:/home/vm # chmod +x bhyvebsdmag.sh
root@akatsuki:/home/vm # vi bhyvebsdmag.sh
root@akatsuki:/home/vm # diff -u /usr/share/examples/
   bhyve/vmrun.sh bhyvebsdmag.sh
--- /usr/share/examples/bhyve/vmrun.sh 2014-02-02
   13:32:01.000000000 +0100
+++ bhyvebsdmag.sh  2014-05-06 23:34:53.000000000 +0200
@@ -32,11 +32,11 @@
 FBSDRUN=/usr/sbin/bhyve

 DEFAULT_MEMSIZE=512M
-DEFAULT_CPUS=2
+DEFAULT_CPUS=1
 DEFAULT_TAPDEV=tap0

-DEFAULT_VIRTIO_DISK="./diskdev"
-DEFAULT_ISOFILE="./release.iso"
+DEFAULT_VIRTIO_DISK="./bsdmag.img"
+DEFAULT_ISOFILE="./FreeBSD-10.0-RELEASE-amd64-bootonly.
   iso"

 usage() {
   echo "Usage: vmrun.sh [-hai][-g <gdbport>][-m
   <memsize>][-d <disk file>][-I <location of installation
   iso>][-t <tapdev>] <vmname>"
root@akatsuki:/home/vm #
```

We start the installation:

```
# ./bhyvebsdmag.sh -i -I FreeBSD-10.0-RELEASE-amd64-
   bootonly.iso bsdmagVM fbsd10
```

Do an install as usual. At the end, don't forget to enter a shell on the installed system (and not directly exit). bhyve does not emulate a graphics card, you need to configure the serial console.

```
# vi /etc/ttys
# cat /etc/ttys|grep console
#     For virtual consoles, the correct type is typically xterm.
# If console is marked "insecure", then init will ask for
   the root password
#console  none              unknown   off secure
console "/usr/libexec/getty std.9600"    xterm   on  secure
# Dumb console
#
```

Then reboot the VM and choose reboot from the beastie menu to exit from the VM.

```
# reboot
```

We are back to our host, now let's start without the install disk and test the VM.

```
root@akatsuki:/home/vm # ./bhyvebsdmag.sh bsdmagVM
```

Wait for boot and login:

```
root@bsdmagVM:~ # uname -a
FreeBSD bsdmagVM 10.0-RELEASE FreeBSD 10.0-RELEASE #0
   r260789: Thu Jan 16 22:34:59 UTC 2014     root@snap.
   freebsd.org:/usr/obj/usr/src/sys/GENERIC  amd64
root@bsdmagVM:~ #
root@bsdmagVM:~ # ifconfig vtnet0
vtnet0: flags=8943<UP,BROADCAST,RUNNING,PROMISC,SIMPLEX,MUL
   TICAST> metric 0 mtu 1500
        options=80028<VLAN_MTU,JUMBO_MTU,LINKSTATE>
        ether 00:a0:98:ac:cf:ce
        inet 192.168.0.111 netmask 0xffffff00 broadcast
    192.168.0.255
        nd6 options=29<PERFORMNUD,IFDISABLED,AUTO_LINKLOCAL>
        media: Ethernet 10Gbase-T <full-duplex>
        status: active
root@bsdmagVM:~ # df -akh
Filesystem       Size    Used   Avail Capacity  Mounted on
/dev/vtbd0p2      23G    2.4G     19G    11%    /
devfs            1.0K    1.0K      0B   100%    /dev
root@bsdmagVM:~ #
```

and some dmesg extract:

```
root@bsdmagVM:~ # dmesg | grep BHY
ACPI APIC Table: <BHYVE  BVMADT  >
acpi0: <BHYVE BVXSDT> on motherboard
root@bsdmagVM:~ # dmesg | grep virtio
virtio_pci0: <VirtIO PCI Network adapter> port 0x2000-
   0x201f mem 0xc0000000-0xc0001fff at device 2.0 on pci0
vtnet0: <VirtIO Networking Adapter> on virtio_pci0
virtio_pci0: host features: 0x1018020 <NotifyOnEmpty,Statu
   s,MrgRxBuf,MacAddress>
virtio_pci0: negotiated features: 0x18020
   <Status,MrgRxBuf,MacAddress>
virtio_pci1: <VirtIO PCI Block adapter> port 0x2040-0x207f
   mem 0xc0002000-0xc0003fff at device 3.0 on pci0
vtblk0: <VirtIO Block Adapter> on virtio_pci1
virtio_pci1: host features: 0x10000044 <RingIndirect,Block
   Size,MaxNumSegs>
virtio_pci1: negotiated features: 0x10000044 <RingIndirect
   ,BlockSize,MaxNumSegs>
root@bsdmagVM:~ #
```

et voila! You have a bhyve VM up:)

Conclusions

As you can see, a little ecosystem starts to grow. And with some tests, you can see and check bhyve with guest as FreeBSD or some other OS.

And please, understand that bhyve needs you, test it and if you think you have something relevant or a bug/crash to show, do consider using the sysutils/panicmail port with bhyve. And don't forget that the best way to speak/exchange about bhyve is the *freebsd-virtualization@freebsd.org* mailing list [8] and the #bhyve channel on Freenode [9].

This is what we are going to see in the second part; we will create VMs for FreeBSD, and some other operating systems to see how it works, through all creation and utilization steps.

Thanks to: Loic Pefferkorn, Laurent Aune, BSDMag team

IMAD SOLTANI
Old school unix/linux lover
http://twitter.com/ximad

References

- *http://bhyve.org/faq/*
- *http://bhyve.org*
- http://bhyve.org/static/bhyve-diagram.jpg [11], [30]
- *https://wiki.freebsd.org/bhyve/pci_passthru*
- *http://2013.asiabsdcon.org/papers/abc2013-P4A-paper.pdf*
- *http://people.freebsd.org/~neel/bhyve/bhyve_instructions.txt*
- *http://people.freebsd.org/~neel/bhyve/vmrun.sh* [1]
- *http://svnweb.freebsd.org/base/head/share/examples/bhyve/vmrun.sh?view=log* [1]
- *https://github.com/grehan-freebsd/grub2-bhyve* [2]
- *http://bsdrp.net/documentation/examples/how_to_build_a_bsdrp_router_lab* [3]
- *http://www.petitecloud.org/docs/hyperv-matrix.jsp* [4]
- *https://git.fedorahosted.org/cgit/virt-manager.git/commit/?id=05df5a64843f2bd4e9a5197d97608d41b2e6dc43* [5]
- *https://github.com/wasted/ezhyve* [6]
- *http://libvirt.org/drvbhyve.html* [7]
- *http://lists.freebsd.org/mailman/listinfo/freebsd-virtualization* [8]
- *http://freenode.net/irc_servers.shtml* [9]
- *http://2014.asiabsdcon.org/papers/abc2014-proc-all.pdf#page=97* [10]
- *http://bhyve.org/presentations/introduction_to_bhyve.pdf*
- *http://bhyve.org/presentations/bhyve-past_present_future.pdf* [12]
- *http://bhyve.org/presentations/bhyve_provisioning_and_monitoring.pdf* [13]
- *http://bhyvecon.org/* [14]
- *http://2014.asiabsdcon.org* [15]
- *http://bhyve.org/presentations/depenguinating_your_infrastructure.pdf* [16]
- *https://github.com/cloudius-systems/osv* [17]
- *http://bhyve.org/presentations/osv_on_bhyve.pdf* [18]
- *http://www.ixsystems.com/whats-new/jenkins-and-bhyve-continuous-integration-for-freebsd-by-craig-rodrigues/* [19]
- *http://2014.asiabsdcon.org/papers/abc2014-proc-all.pdf#page=07* [20]
- *http://people.freebsd.org/~grehan/talks/vendor-summit.pdf#page=4* [21]
- *http://bhyve.org/presentations/openfest2013-handout.pdf* [22]
- *http://www.bsdnow.tv/tutorials/bhyve*
- *http://2013.asiabsdcon.org/papers/abc2013-P5A-paper.pdf* [23]
- *http://www.freebsd.org/cgi/man.cgi?query=tmux&sektion=1&apropos=0&manpath=FreeBSD+9.0-RELEASE+and+Ports* [24]
- *http://www.freebsd.org/cgi/man.cgi?query=screen&apropos=0&sektion=1&manpath=FreeBSD+9.0-RELEASE+and+Ports&arch=default&format=html* [25]
- *http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/book.html#virtualization-host-bhyve* [26]
- *http://www.bsdstore.ru/en/about.html* [27]
- *http://bhyve.org/presentations/openfest2013-handout.pdf* [28]
- *http://standards.ieee.org/develop/regauth/oui/public.html* [29]

Getting to Grips with the Gimp – Part 4

In the fourth part in our series on the Gimp we will learn about guides and paths.

What you will learn…

- How to manipulate images like a design pro

What you should know…

- General PC administration skills

One of the most powerful tools supplied by the Gimp is the paths tool. Using Bezier curves, selections can be made accurately around curved objects, and minor adjustments and transformations performed on the selection. This selection can be saved as a path and applied across layers, making this function a great time-saver when working with complex images. While Bezier curves can be difficult to master, with practice they will become an essential part of your toolkit. Depending on the image you want to manipulate, Beziers can be a hindrance though. Sometimes it is quicker just to select the outline manually. We will demonstrate both in this tutorial, and end up with a controversial image with a text flowing to a path.

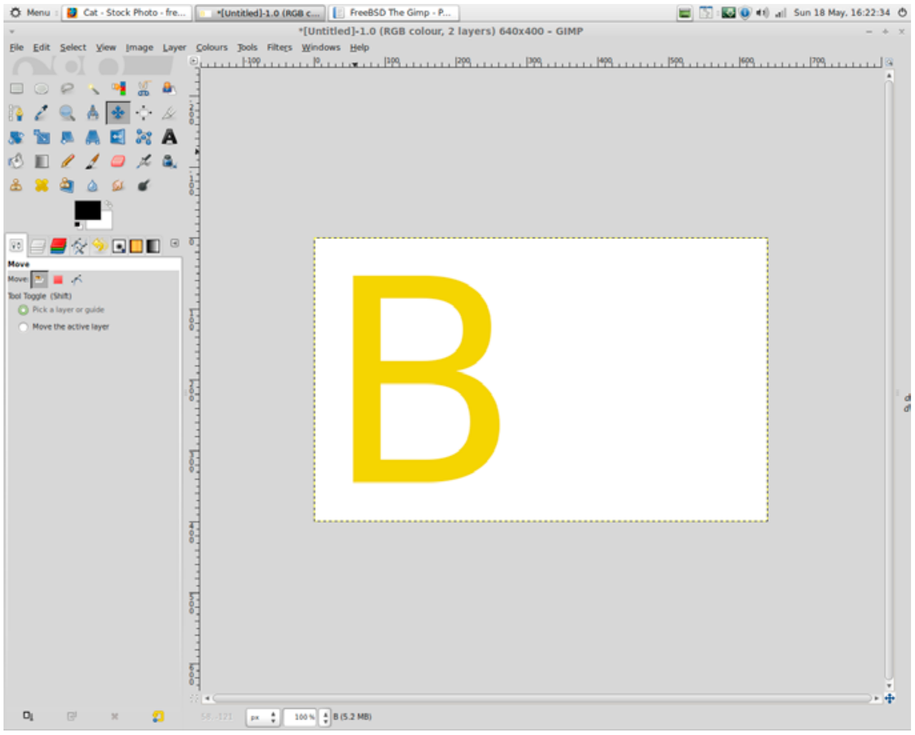
Lets get started!

Greed is Good! Fat cats of the world unite,

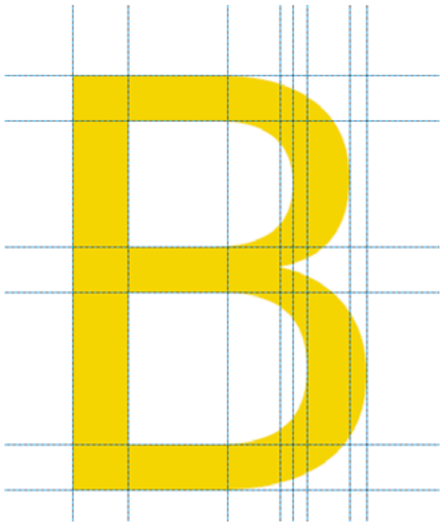
Part 1 – Learning Bezier curves and paths

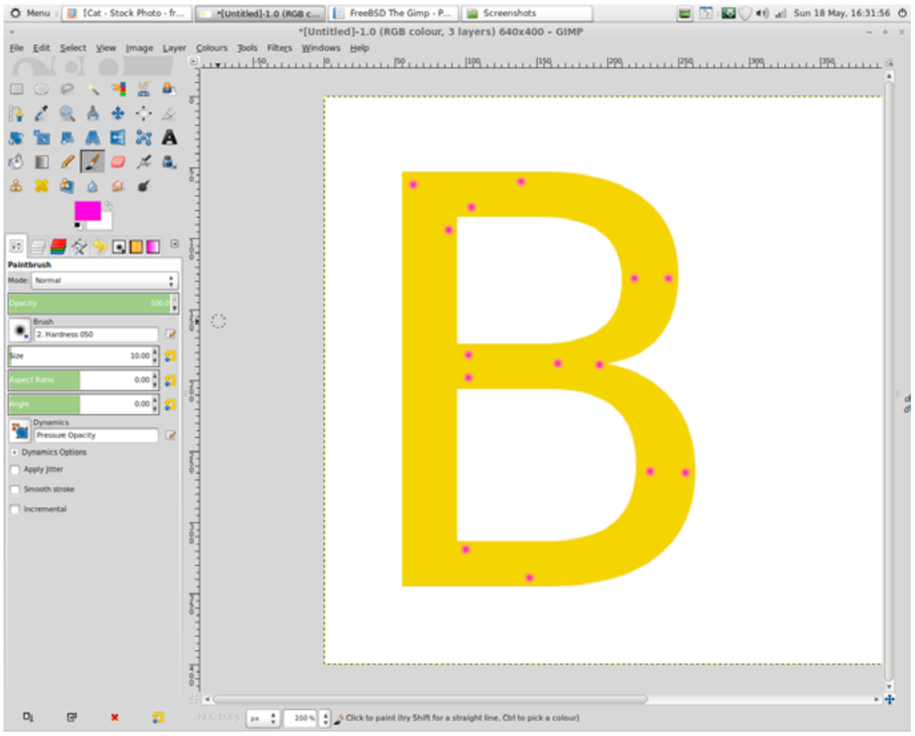
Step 1
Create a new image 640 x 480 px.

Step 2
Using the text tool, create the letter "B" in pale yellow 400px high. Layer to image size [Figure 1].



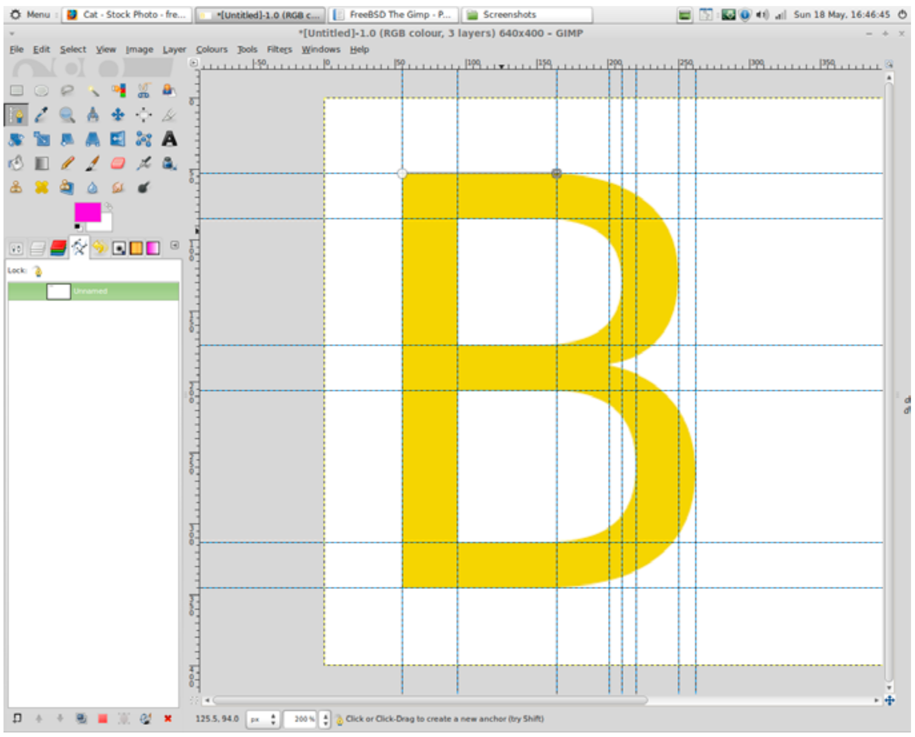
Step 3
Click and drag guides from the horizontal and vertical measuring ruler to the intersection points marked in magenta [Figure 2]. Zoom in if required to get accurate placement, resulting in the screenshot shown in Figure 3.



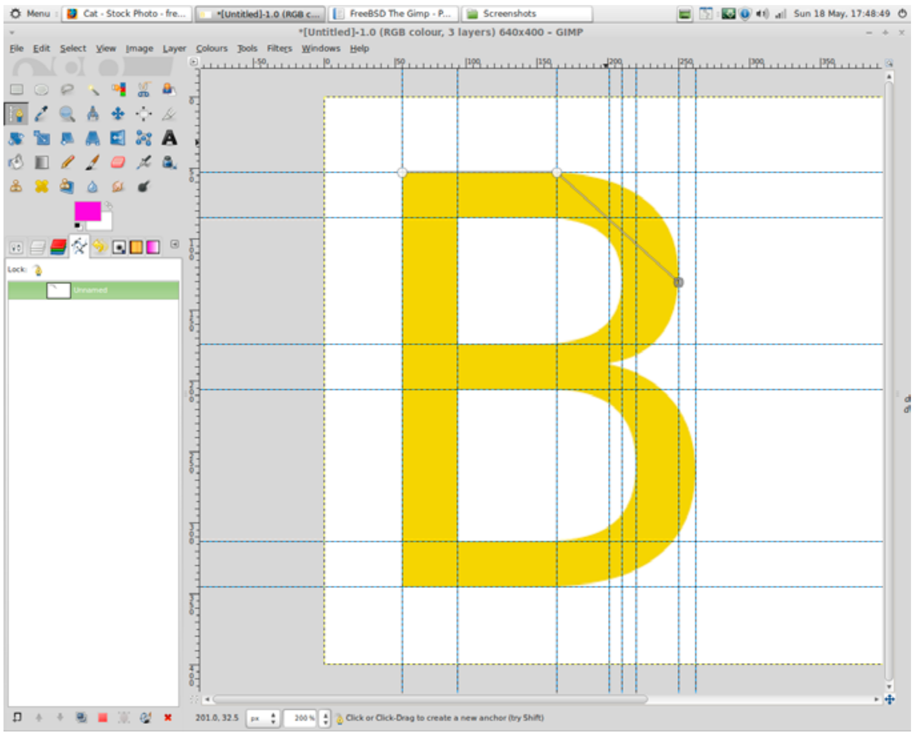


Step 4
Click on the paths tool and, starting at the top left hand side of the "B", click once and a small square and circle will appear. If you click and drag on the middle of the icon, you can easily re-position the center point. Note how the center of the selection will snap to the guides. Move the selection point back to the top left hand side of the "B".

Step 5
Click once on the 3rd vertical guide on the letter B which will result in a straight path [Figure 4].


*[Untitled]-1.0 (RGB colour, 3 layers) 640x400 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
125.5, 94.0
Click or Click-Drag to create a new anchor (try Shift)

Step 6
Following the 7th vertical guide to the first upper curved bulge of the "B", click once. This will result in a straight path bisecting the 4th vertical guide and the second horizontal guide [Figure 5].


*[Untitled]-1.0 (RGB colour, 3 layers) 640x400 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
201.0, 32.5
Click or Click-Drag to create a new anchor (try Shift)

Among clouds Performance and Reliability is critical

Download syslog-ng Premium Edition product evaluation here

Attend to a free logging tech webinar here

BalaBit IT Security

www.balabit.com

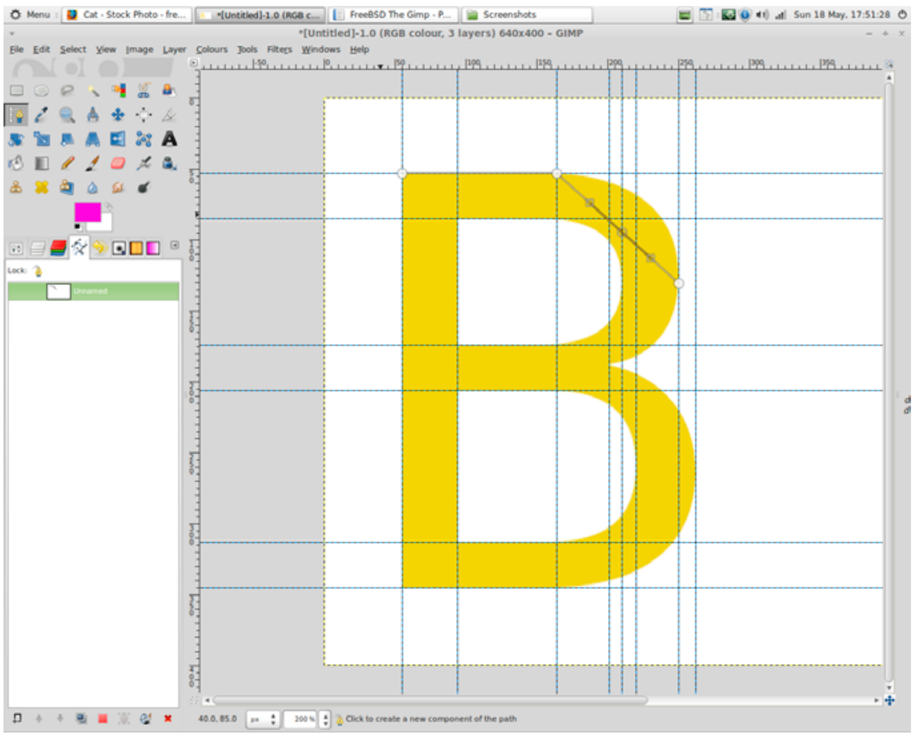
syslog-ng log server

The world's first High-Speed Reliable Logging™ technology

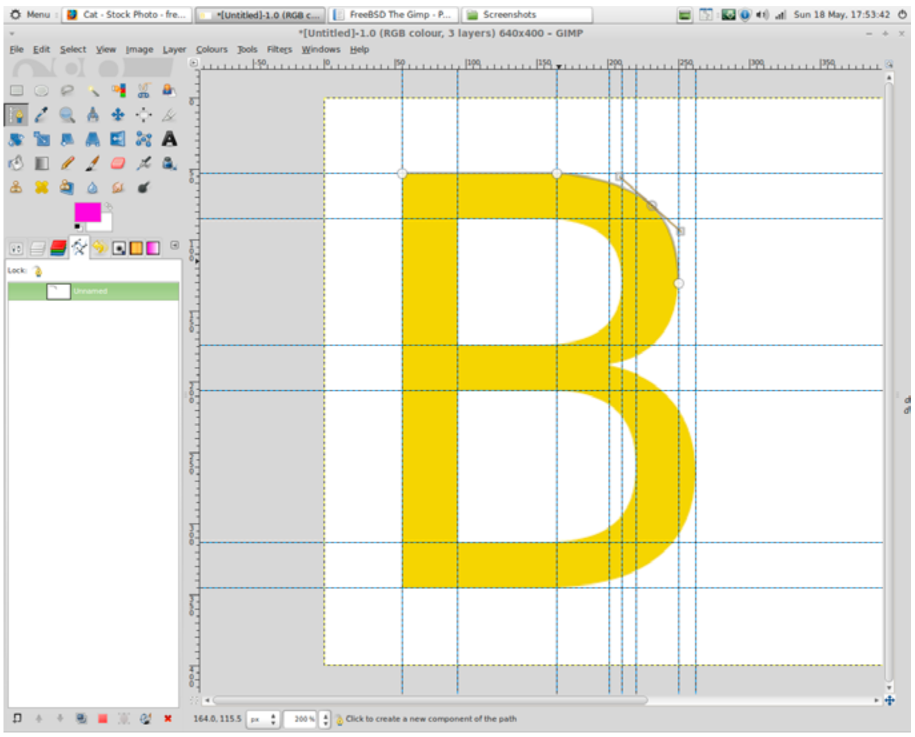
HIGH-SPEED RELIABLE LOGGING

- above 500 000 messages per second
- zero message loss due to the Reliable Log Transfer Protocol™
- trusted log transfer and storage

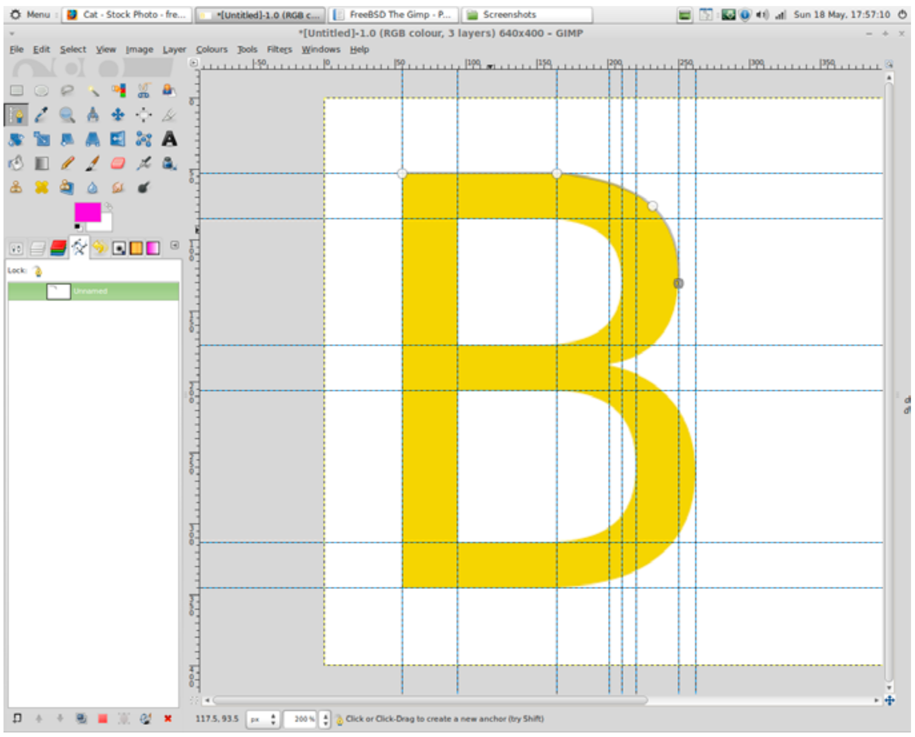
Step 7
Holding the Ctrl key, click on the path where it intersects with the 5th vertical guide [Figure 6].


*[Untitled]-1.0 (RGB colour, 3 layers) 640x400 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Click to create a new component of the path

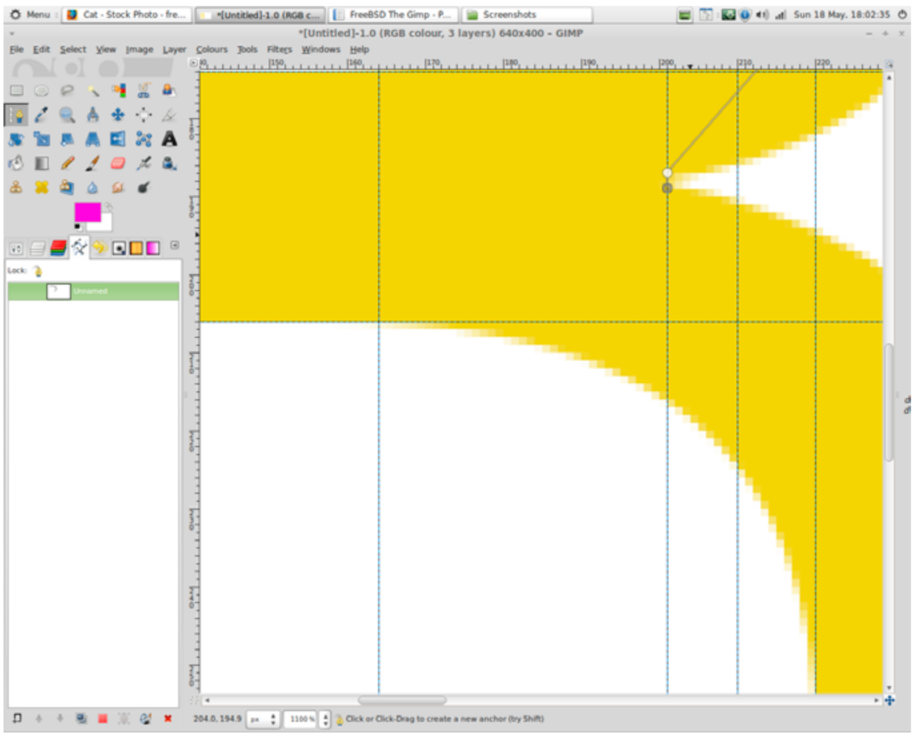
Step 8
Click and drag the center point of the node so it follows the curve of the "B" [Figure 7].


*[Untitled]-1.0 (RGB colour, 3 layers) 640x400 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Click to create a new component of the path

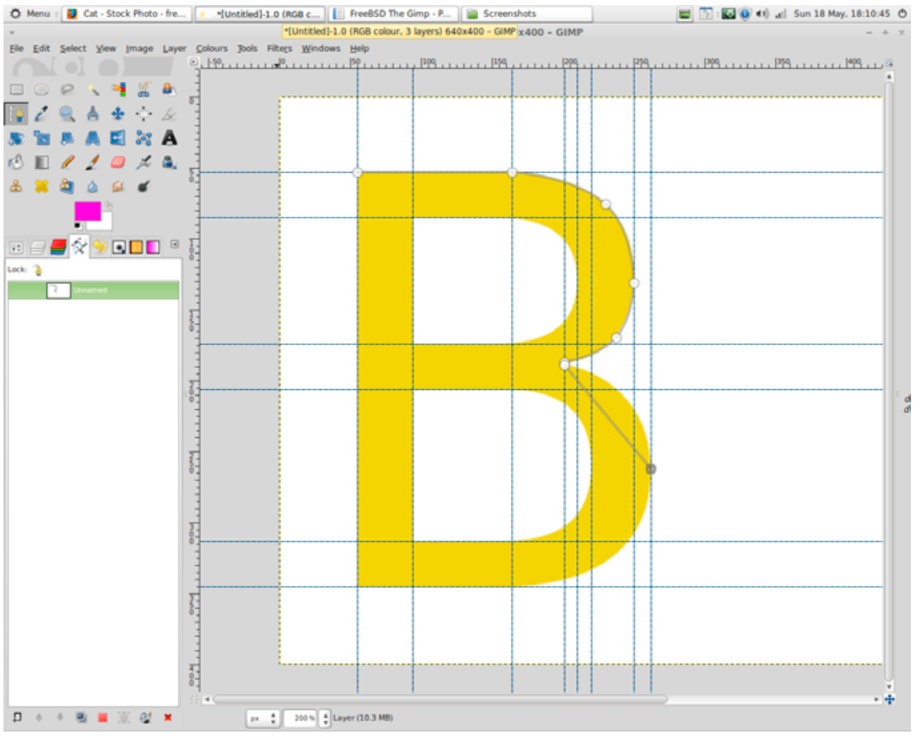
Step 9
Click on the end node so that a square appears [Figure 8].



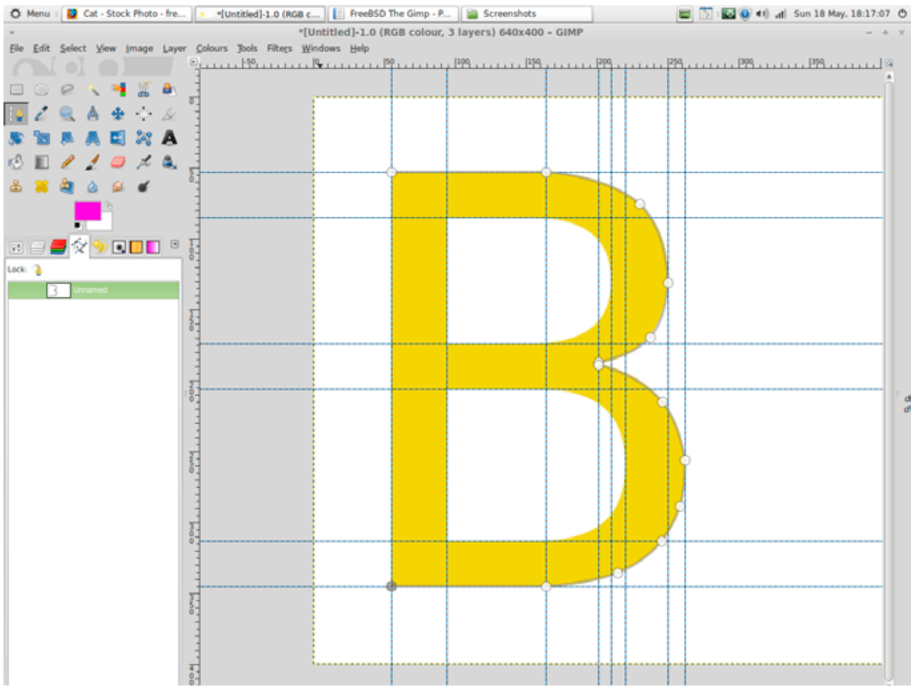
Step 10
Press the + key to zoom in (Do not use the Zoom tool as you will lose your path) and add 2 nodes where the two upper and lower curves of the letter merge [Figure 9].



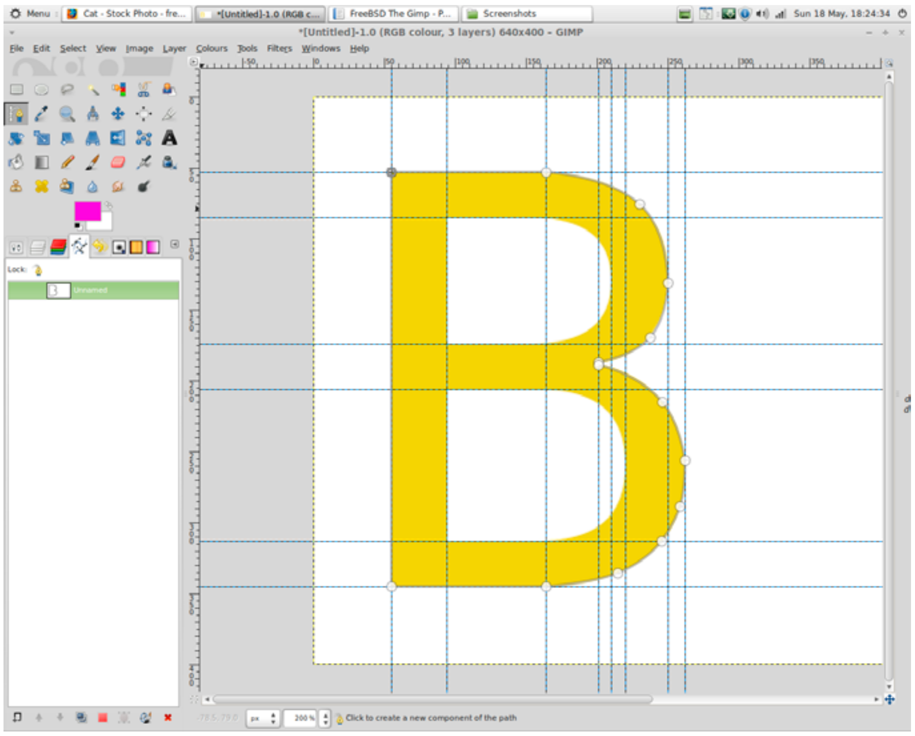
Step 11
Zoom out using the – key on the keypad. Repeat steps 7-9 with the lower part of the upper curve and the upper part of the lower curve [Figure 10].


*[Untitled]-1.0 (RGB colour, 3 layers) 640x400 - GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Sun 18 May, 18:10:45

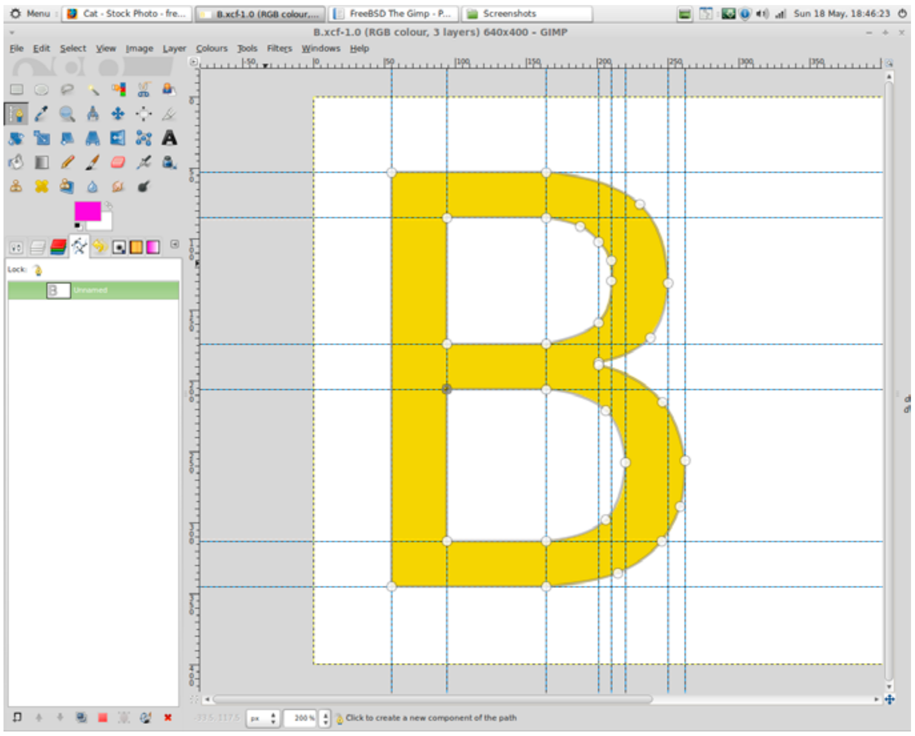
Step 12
Carry on adding nodes and adjusting the center point of the nodes until you reach the bottom left hand side of the "B" [Figure 11].


*[Untitled]-1.0 (RGB colour, 3 layers) 640x400 - GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Sun 18 May, 18:17:07

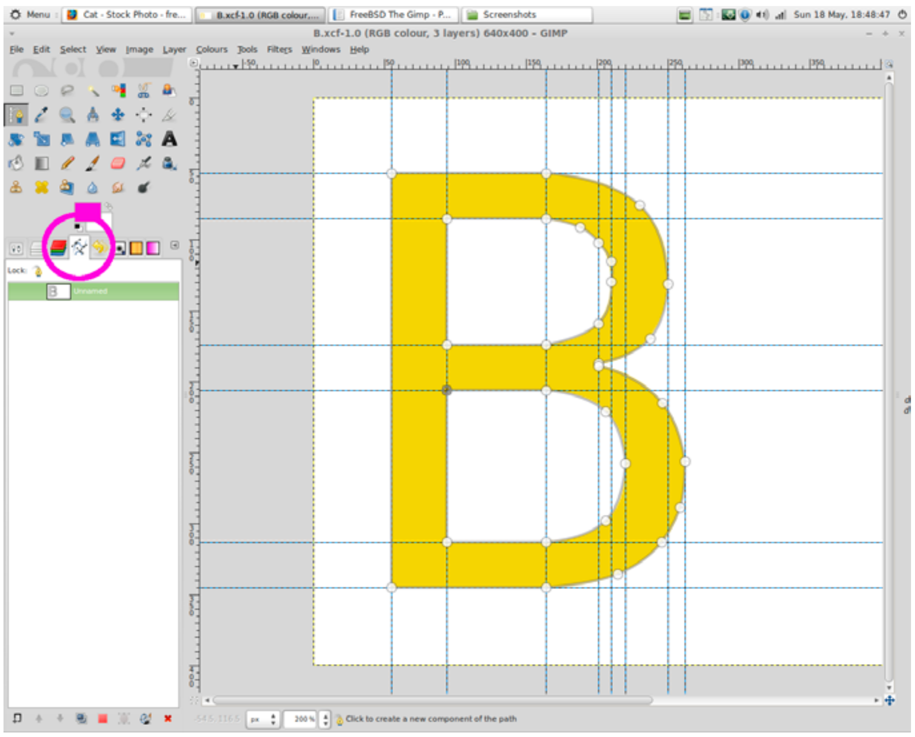
Step 13
Press and hold the Ctrl key and the cursor will change to a union symbol when you hover over the node created in Step 4. Click to join up the path [Figure 12].


Menu
Cat - Stock Photo - fre...
*[Untitled]-1.0 (RGB c...
FreeBSD The Gimp - P...
Screenshots
Sun 18 May, 18:24:34
*[Untitled]-1.0 (RGB colour, 3 layers) 640x400 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Lock:
Unnamed
200 %
Click to create a new component of the path

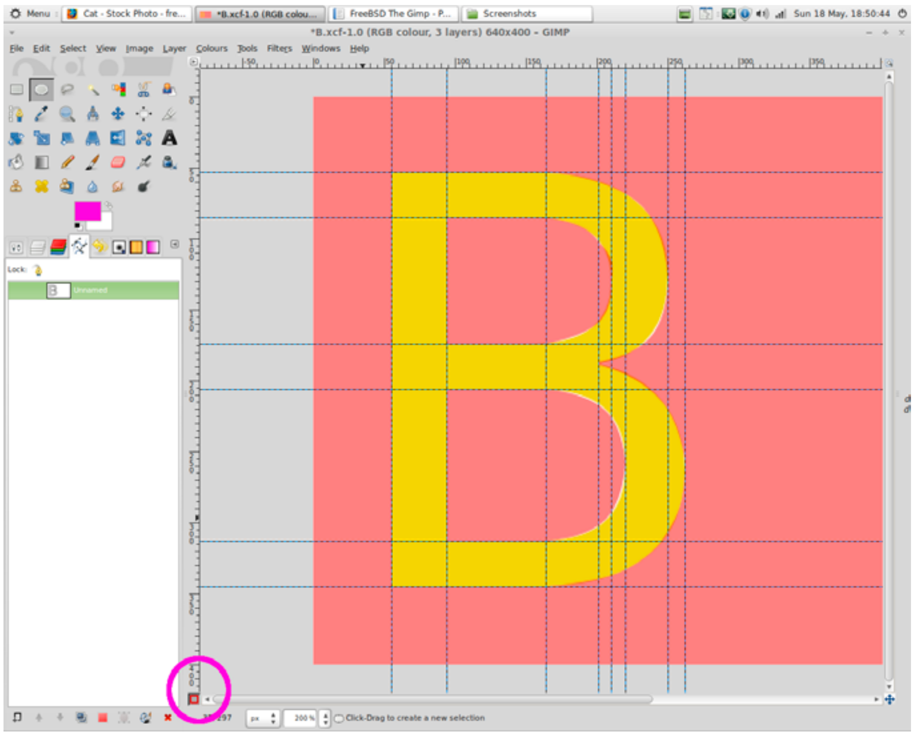
Step 14
Repeat steps 4-13 with both inner areas of the "B". Don't forget to click on the end node before attempting to create another, otherwise you will have an orphan node. If this happens, press Ctrl Z to recover. [Figure 13].


Menu
Cat - Stock Photo - fre...
B.xcf-1.0 (RGB colour,...
FreeBSD The Gimp - P...
Screenshots
Sun 18 May, 18:46:23
B.xcf-1.0 (RGB colour, 3 layers) 640x400 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Lock:
Unnamed
200 %
Click to create a new component of the path

Step 15
Click on the paths tab to see an outline of the final path [Figure 14].


B.xcf-1.0 (RGB colour, 3 layers) 640x400 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Click to create a new component of the path

Step 16
Right click on the path and choose Path → Selection. Click on the quick mask button at the bottom left hand side of the canvas and you will see the masked off areas [Figure 15].


*B.xcf-1.0 (RGB colour, 3 layers) 640x400 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Click-Drag to create a new selection

Developing for Amazon Web Services?

Attend Cloud DevCon!

Cloud DevCon

June 23-25, 2014

San Francisco

Hyatt Regency Burlingame

www.CloudDevCon.net

Attend Cloud DevCon to get practical training in AWS technologies

- Develop and deploy applications to Amazon's cloud
- Master AWS services such as Management Console, Elastic Beanstalk, OpsWorks, CloudFormation and more!
- Learn how to integrate technologies and languages to leverage the cost savings of cloud computing with the systems you already have
- Take your AWS knowledge to the next level – choose from **more than 55 tutorials and classes,** and put together your own custom program!
- Improve your own skills and your marketability as an AWS expert
- Discover HOW to better leverage AWS to help your organization today

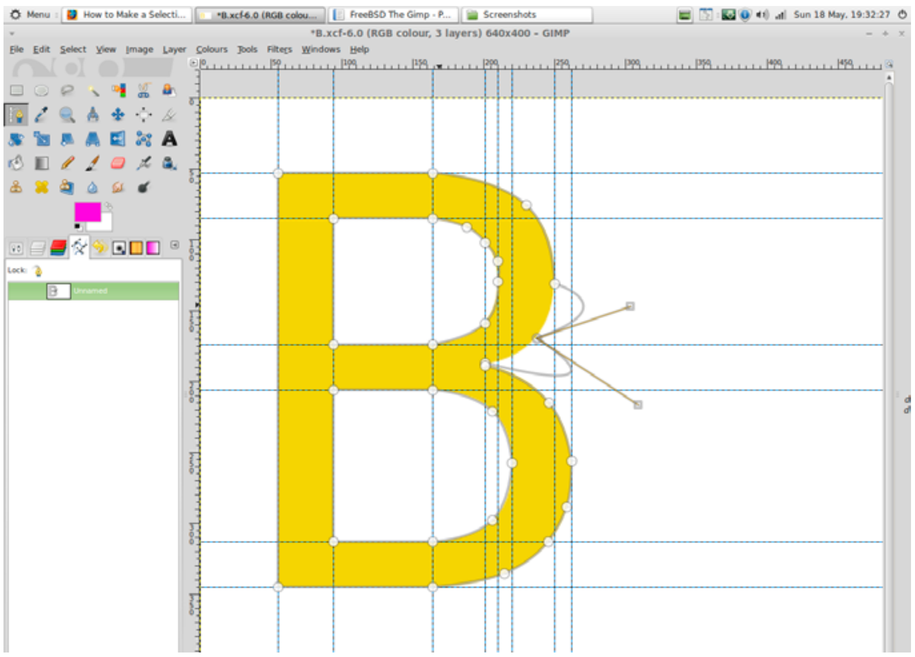
Register Early and SAVE!

A BZ Media Event

CloudDevCon

Step 17

Un-click the quick mask and right click on the path dialog and choose path tool. Note that when you click on a point on the curve, adjustment "arms" appear that will allow you to adjust the curve once it has been created, rather than just moving the center node [Figure 16].


*B.xcf-6.0 (RGB colour, 3 layers) 640x400 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help

Part 2 – Fat cat logo

Step 1

Download the images from Table 1.

Table 1. *Details and credits*

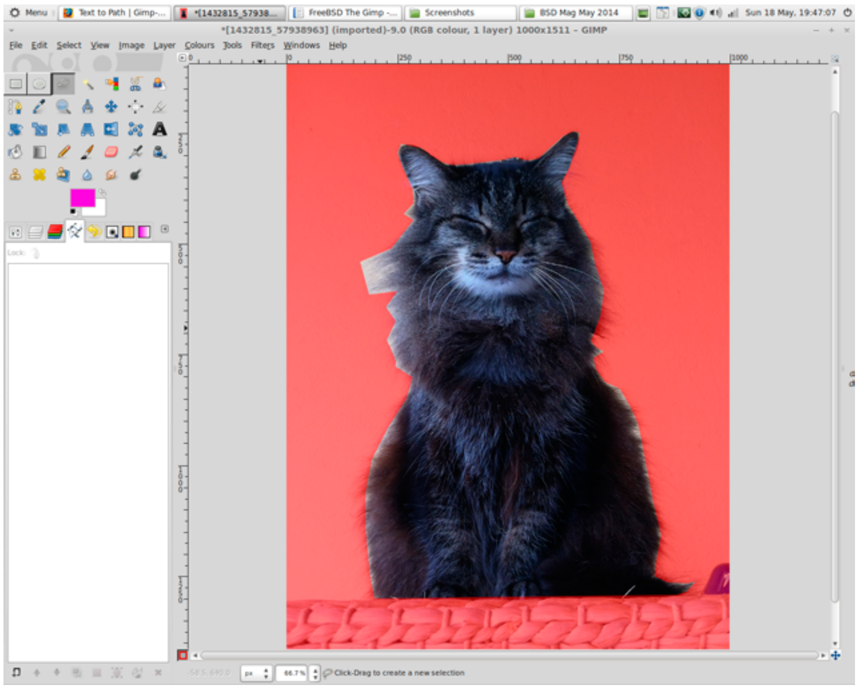
Image	URL	Details and credits
Cat	*http://www.freeimages.com/photo/1432815*	Closed eyes cat. Uploaded by diogoakio
Bengal cat	*http://www.freeimages.com/photo/1435180*	Bengal Cat on Blanket Bengal Cat lying on white Blanket in the Garden, just woke up Uploaded by Krappweis

Step 2

Open the Cat image (Thanks, *diogoakio!*) and using the free select tool, select a rough outline of the cat. The fine hairs on the cat are not too important, so don't spend too much time on that part on the detail [Figure 17 with mask enabled to show detail].

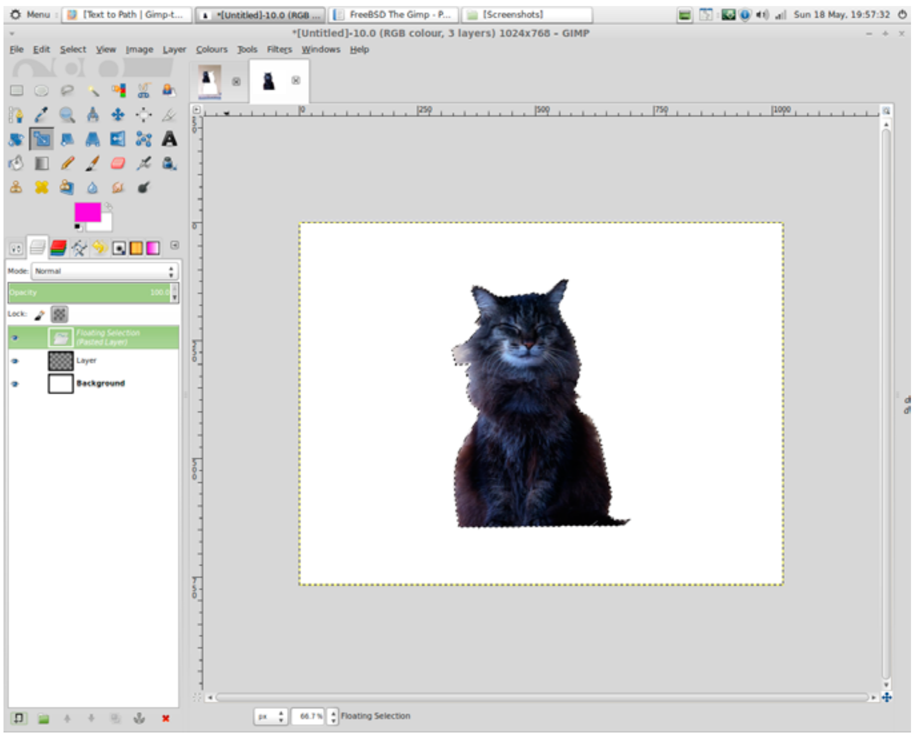
Step 3

Click on the scale tool and ensure both the X and Y axis is constrained. Scale to 50%, and click on Edit → Copy.


*[1432815_57938963] (imported)-9.0 (RGB colour, 1 layer) 1000x1511 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Click-Drag to create a new selection

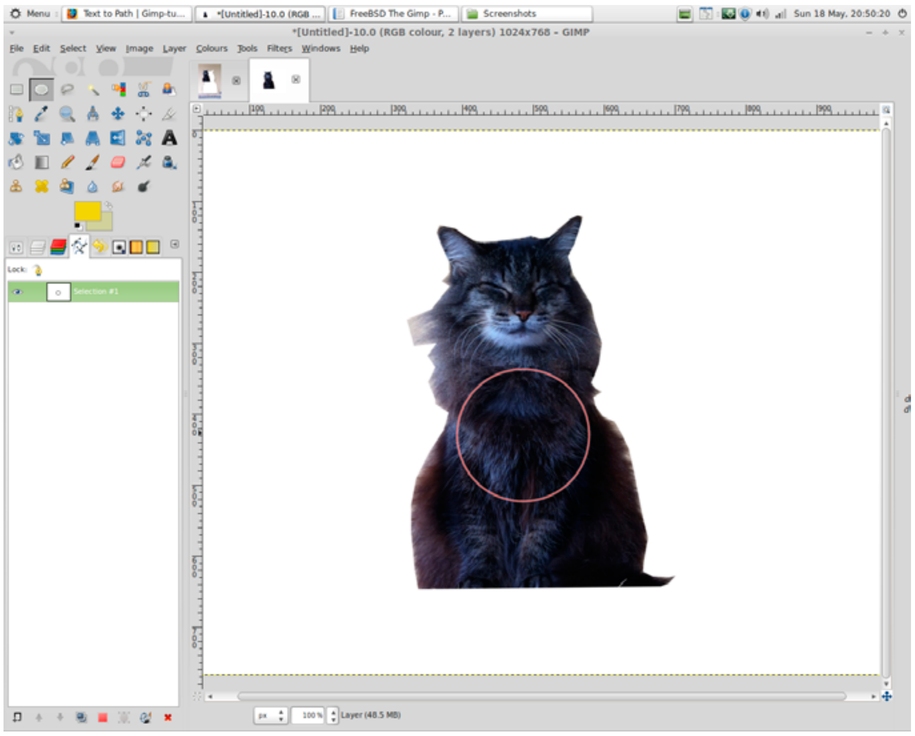
Step 4

Create a new image 1024 x 768 px from File → New, add a transparent layer and click on Edit → Paste [Figure 18].

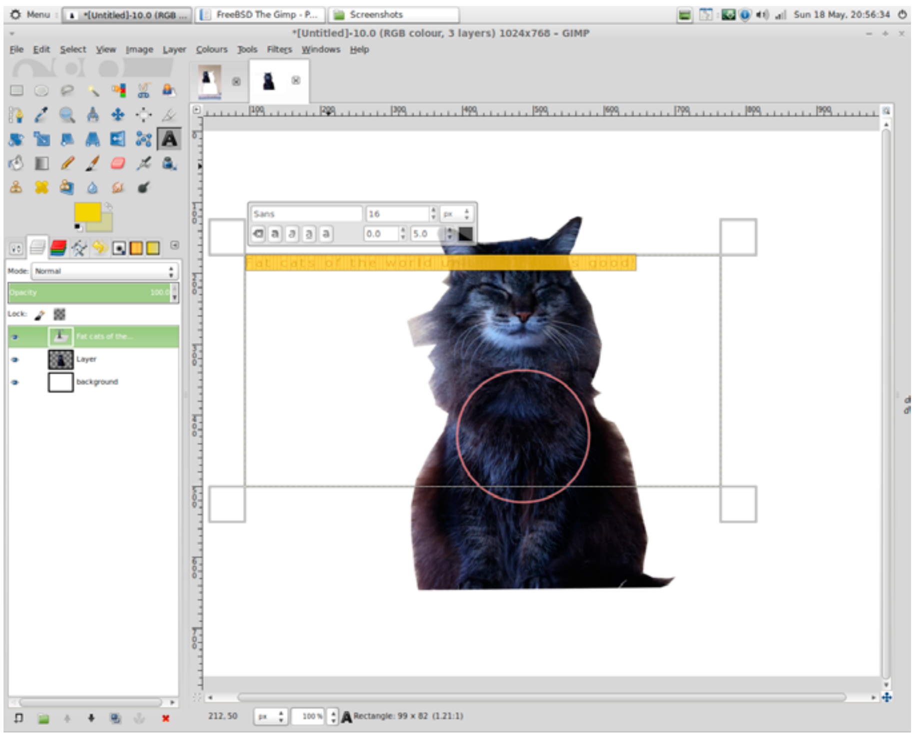

*[Untitled]-10.0 (RGB colour, 3 layers) 1024x768 - GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Mode: Normal
Opacity
Lock:
Floating Selection (Pasted Layer)
Layer
Background
Floating Selection

Step 5

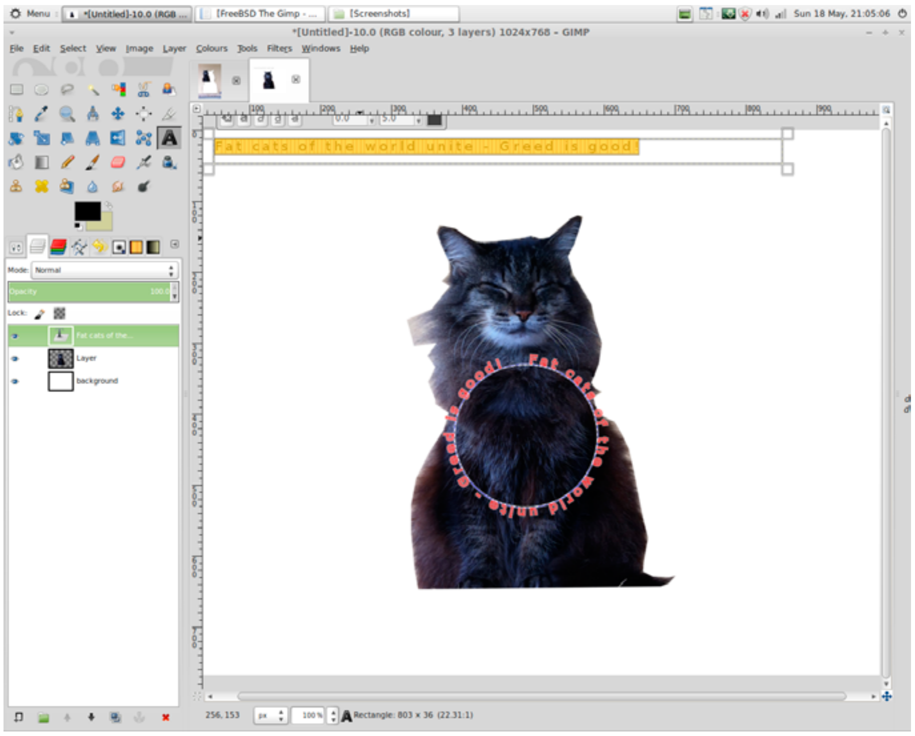
Anchor the layer, add another layer and using the ellipse select tool, create a circle around the middle of the cat using Shift to constrain. Right click and Select → to path. Click on the paths tag, click on the eye and you will see the circular path [Figure 19].


*[Untitled]-10.0 (RGB colour, 2 layers) 1024x768 - GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Lock:
Selection #1
Layer (48.5 MB)

Step 6
Create the text “Fat cats of the world unite – Greed is good!” sized 16px Sans with a kern of 5px [Figure 20].

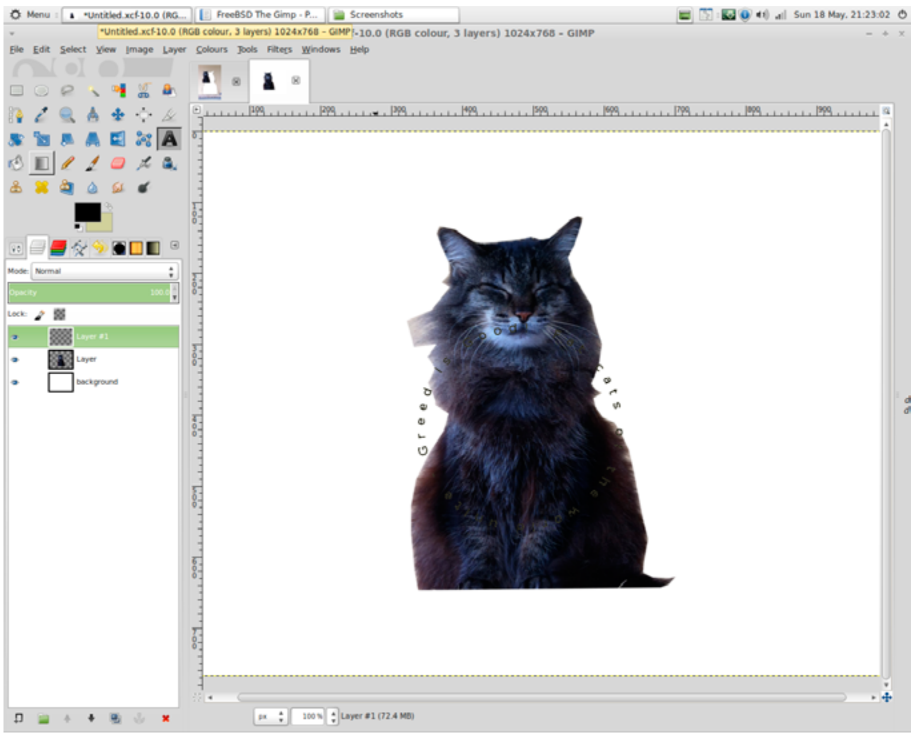


Step 7
Click on the layers tab and right click select → Text along path. Remove the text layer [Figure 21].



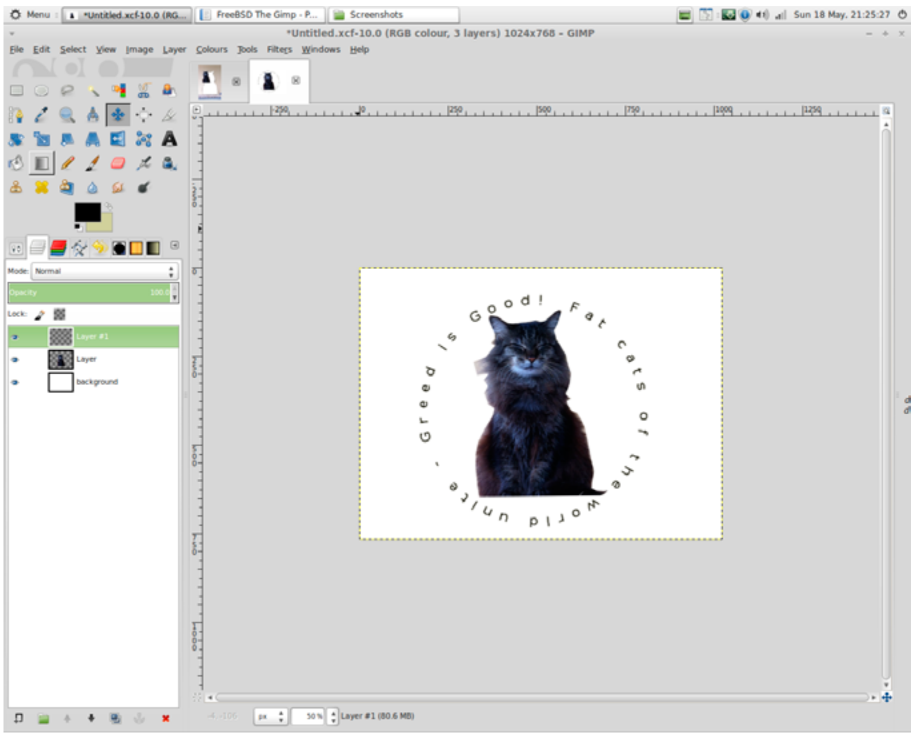
Step 8

Return to the paths tab, select Path → Selection and press Ctrl + . or Ctrl +, to fill with the background or foreground color as desired. From the menu, Select → None [Figure 22].

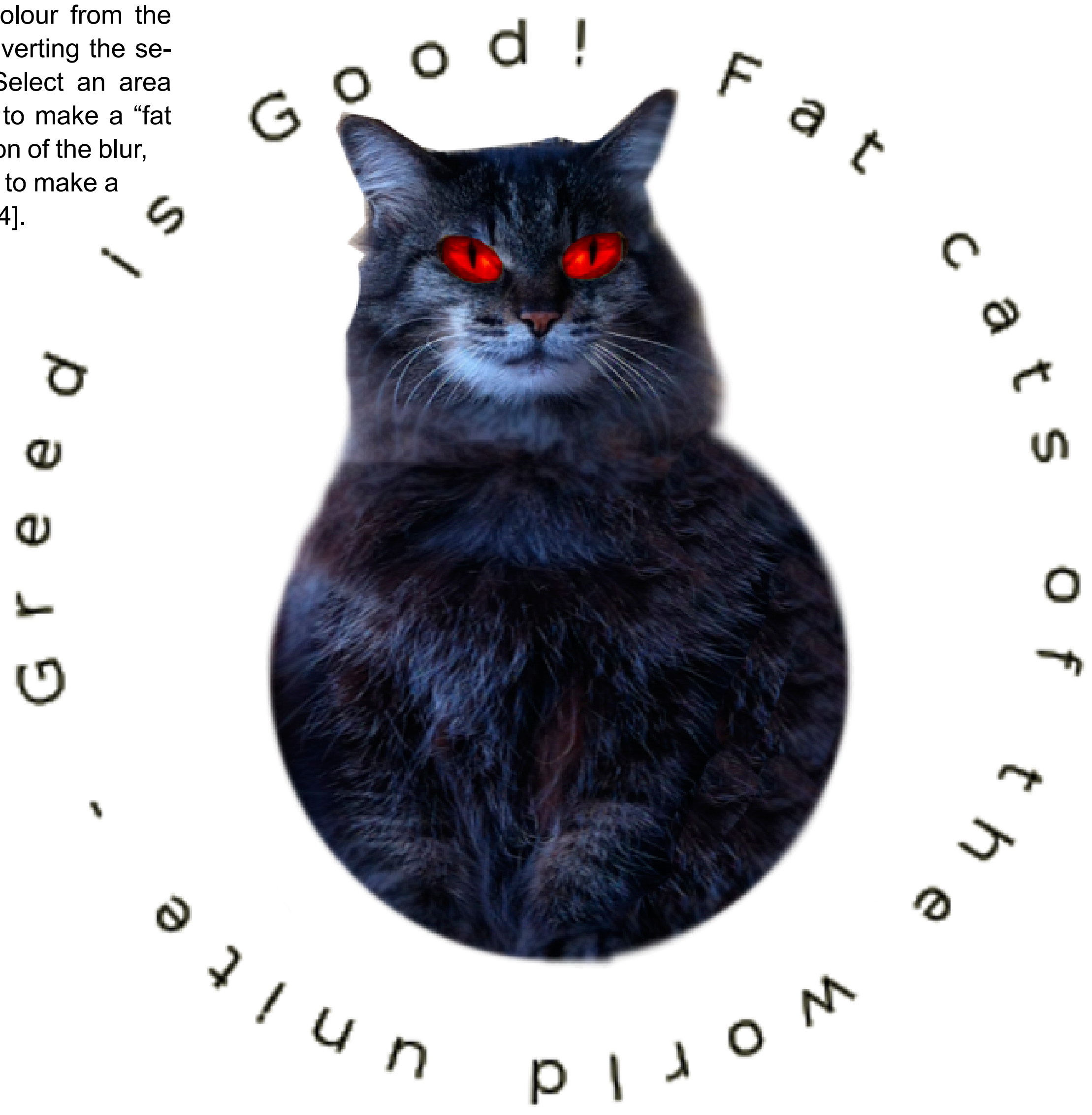

*Untitled.xcf-10.0 (RGB colour, 3 layers) 1024x768 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Mode: Normal
Opacity
Lock:
Layer #1
Layer
background
Layer #1 (72.4 MB)
Greed
cats

Step 9

Using the scale tool, stretch the contained layer to wrap around the cat [Figure 23].


*Untitled.xcf-10.0 (RGB colour, 3 layers) 1024x768 – GIMP
File Edit Select View Image Layer Colours Tools Filters Windows Help
Mode: Normal
Opacity
Lock:
Layer #1
Layer
background
Layer #1 (80.6 MB)
Greed is Good! Fat cats of the world unite

Step 10
Copy and paste the eyes from the Bengal cat and add a red layer to colour. Remove any colour from the rest of the image by inverting the selection and deleting. Select an area of the body and scale to make a "fat cat", using a combination of the blur, smudge and erase tool to make a realistic body [Figure 24].



While the resulting demo in this article isn't perfect, time and perseverance will improve it!

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.

Faster.
Better.
Reliable.

Trusted by over 500 ISPs worldwide.

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

Features:

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

hyper

Remote Install

Using your hardware

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

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

Interview with Luca Ferrari

Luca, please introduce yourself to our readers.

I have a Computer Science degree and PhD and I have been an Adjunct Professor at Nipissing University. I was involved with the Italian PostgreSQL Users' Group (ITPUG) from the very beginning, being a co-founder, vicepresident and for the past year the current president. In my daily job I develop and maintain web based applications. I advocate Free and Libre OpenSource Software, that is in my opinion the only rational way of doing software. I am always interested in operating systems development and technologies and in new programming languages, with particular regard to the scripting ones.

I live in Italy with my beautiful wife and lovely son.

Could you tell us more about your background?

My very first computer was a Commodore-64, a gift from my father who has always been better than me at pretty much every game we challenged. During the high-school I learnt the C language, it was the Microsoft DOS age. I remember even doing some trivial archery games using the graphic libraries available back then. The shift to *nix happened during the University years, when I was first introduced to Unix and Linux. It took me some time to fall in love with the shell and the power of the *nix tools, and for a few years I was using only Linux as my main desktop and development machine. I began self-learning C++ and Java, finding the former better even if complex. I then improved my skills with Perl, a language that greatly influenced my culture and programming style. With the shell, Perl and C in my hands, I did the complete transition to Linux and began building my own set of utilities for automated and remote backup, reporting, etc. I have to say that I switched to Linux simply because it was the operating system installed in the University laboratories, and so it appeared to me as the only alternative to commercial Unix distributions. Of course, I was wrong! However, I believe this is an important point against the spread of BSD: a lot of schools and universities only present the Linux-alternative, without any mention of BSDs.

During my PhD at the University, I was in charge of a few Sun Ultrasparc II workstations, and that was my very first experience with a real Unix OS (Solaris). I was not really impressed by this experience, since keeping Solaris up- to-date and well configured was quite complex for me, but I was strongly influenced by the Linux way.

Once finished at the University I had the luck of finding a job as a Linux system administrator, and therefore I was able to improve my skills day by day on a real use case. It was during this job that I started feeling uncomfortable with Linux systems: having to deal with a lot of servers with different distributions, patch levels, and packet managers was quite a mess. I am not saying that it is not worth using it, and in fact I still run a set of Linux systems, but I was simply searching for something more accurate and coherent to ease the management of multiple servers. I then started learning *BSD, and progressively switched to OpenBSD and FreeBSD (depending on the specific use case). Having to work also with non-sysadmins I found it very important to have built-in variants like pfSense and FreeNAS that allowed me to work with all the power of a real *BSD and enabled my coworkers to manage the device as a black-box.

During my switch to *BSD I also began working as a Database Administrator for small to medium sized relational databases. Being an OpenSource addict I started studying PostgreSQL, and deploying it in production environments.

Please tell us about your proudest achievements?

From a computer science point of view they are surely my PhD, dated 2003, and the position of Adjunct Professor at Nipissing university, held from 2011 to 2014. I am proud of my PhD because it taught me a lot about how to do, and how not to do, research. After I left the University I continued doing research and producing some papers, and so and this is how I gained the Adjunct Professor position.

On the OpenSource side, I am really proud of my involvement in the ITPUG and being voted as president in 2013 was an important achievement that demonstrated the time and efforts I put into the association were worth it.

Despite the computer science, my greater achievements were in archery back when I was teenager.

Please tell our readers, what the *BSD OSes future looks like?

Well, these days the BSD operating systems are a step ahead of every other operating system. Just look at the

BSD Certification

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

WHAT CERTIFICATIONS ARE AVAILABLE?

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

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

WHERE CAN I GET CERTIFIED?

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

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

WHERE CAN I GET MORE INFORMATION?

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

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

great work done by the OpenBSD team on 64 bit compatibility or security, or consider the integration of ZFS and the new virtualization system (Bhyve) into FreeBSD, or the number of platforms on which NetBSD can run. And moreover a BSD system is no longer scary: we have PCBSD that has done a lot of work in terms of ease of use and installation; web based distributions like pfSense, FreeNAS and NAS4Free, and the others emerging. I believe BSD will be an important part of the future, as well as it has played a very fundamental role in the past.

Could you tell more what the best capabilities of *BSD are from your point of view.

Stability and reliability are surely the most attractive features of pretty much every major *BSD out there. Documentation is another very important aspect of almost every *BSD: it is very accurate and always available unlike other operating systems. Documentation allows every sysadmin to learn more and improve herself. Finally, another very important feature I like in the *BSDs is the coherence of the whole system, that is shipped in “base” as a full system and not as a puzzle of several parts to plumb together. This means that I can always trust that what is in base will work seamlessly.

What was/is your best tool to work with?

GNU Emacs. It is surely the application I spend most of my daily work time in. Second comes my shell, that these days defaults to zsh. And then comes a few different applications that are available on my favorite desktop: KDE. As a programming language, I simply love Perl, even if I cannot say why, as well as C, but in my everyday job I use Java and PHP. As a relational database I use PostgreSQL every time I can. Finally, I use Linux at work, but I love FreeBSD as an operating system and I use it at home and everywhere I can.

What is the best advice for those who want to use the *BSD OS and why should they?

First of all, everyone interested in computing should try a *BSD at least one time in her own life. The reasons are quite simple and well known:

- they are free systems and do not impose a vendor lock-in;
- they will force users to learn what they are doing, meaning that users will progressively take control over their own computers, instead of being mere users;
- they are well documented and there are communities to get support from (and besides, people will learn how to ask for help in the right way);
- they work. It does not matter if you are running your own desktop, an at-home firewall, a small office NAS or a super-cluster, *BSDs just work.

I strongly encourage people to try and learn *BSDs because they represent a great opportunity to learn something new or to consolidate their own skills on a set of platforms that are widely adopted. These systems are OpenSource ones, and therefore everyone using them has the chance to learn and improve herself.

As a general advice I believe that once you want to try a new system you have to commit to it entirely: only facing day by day problems and solutions will make you learn and improve on such system. These systems are professional, made by professionals for other professionals. So become a professional as well.

How do you want to improve yourself in the next year?

I do not have specific plans for the next year, but I try every year to participate in conferences and meetings to keep myself up-to-date and get new ideas and projects worth looking at. Besides this, I read a lot of books and papers and experiment myself in order to learn even the new features I will seldom use.

Another huge source for learning is the source code of many applications and systems, and today it is quite easy to follow the development thanks to a lot of distributed revision control systems (e.g., git).

What *BSD systems do you use?

I use FreeBSD the most, considering also that I run FreeNAS and pfSense that are based on the former; I also use PCBSD for family members and friends. When not using FreeBSD I use OpenBSD, which I consider one of the most important efforts in the OpenSource space.

Thank you,
Ewa & BSD Team

UPDATE NOW WITH STIG AUDITING

"IN SOME CASES **nipper studio** HAS VIRTUALLY **REMOVED** the NEED FOR a **MANUAL AUDIT**"

CISCO SYSTEMS INC.

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

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

2012 Computing Security Awards WINNER Enterprise Security Solution of the Year

Security Products' GOVIES 2013 Government Security Awards

2012 Computing Security Awards WINNER Network Security Solution of the Year

www.titania.com

In a Surprise Decision, Europe's Top Court has Ruled That Google can be Forced to Erase Links to Content About Individuals. Is History Repeating Itself Once Again?

Those who cannot remember the past are condemned to repeat it, or so said the philosopher George Santayana. In a previous article I had a good rant on how the UK Conservative coalition – during an unexpected but not unsurprising period of hypocrisy – requested the removal of all of the parties' election promises from the Internet while at the same time supported the banning of over 100K keywords related to child pornography. To reiterate, I find both actions morally repulsive, because a) the former should be a permanent matter of public record and b) the latter is as useful as a chocolate teapot in defending children from the evil of pedophile abuse. For those that are still listening, generally, technology is not best served by committees, lawyers, judges, politicians or indeed governments as they just "don't get it". That is not to say that there are not IT savvy individuals in these sectors – there are – but generally their voices are drowned out by the herd of the ignorant.

Fairly soon they will take me away in a wooden box and bury me in the ground. Whether that be 5 years, 10 years or 25 years from now matters not one whit, but I will go to my grave ranting that what is happening to the Internet and the World Wide Web today is identical to what happened during the time in history when books first became the "new media". Ergo, the establishment got pretty worked up about the peasants finding out "esoteric knowledge" and the church (in this case the Roman Catholic Church) stepped in to be "God's censor". The fact that this knowledge was the Protestant bible (and there are those who argue the protestants didn't go far enough) is neither here nor there – the principle remains that an element of the establishment was not happy and did everything in its power to censure, burn, character assassinate and torture those on the side of an alternative view, revelation and freedom of speech.

I am not a great defender of corporate culture – particularly corruption and ethical misdeeds, and it is rare that I will be found taking the side of Google – the new Microsoft. But in this case, they have a point. If Google is to be taken seriously and if you are to look at today's younger generation, Google is indeed the font of all wisdom – how can redacting the past elicit credibility? And herein lies the problem – is Internet content a pastiche, a snapshot of history or just a temporary communication medium without value? As the old saying goes, a verbal contract is not worth the paper it is written on. After all, Google is the world's biggest log-file.

Where I come from, a man's word is his bond, and I expect the same from my computer. A trusted and proven friend will have more credibility than a stranger, likewise there are a number of websites I trust. While Google is still fighting the blowback from its original ethical stance of "do no evil" in the UK due to ethical issues over tax, it is still the world's leading search engine. There are those that can add to this faux pas, and no doubt the underbelly of Google will be found to be as grubby as any modern corporation. However, that should not divert us from this ruling that gives pedophiles, criminals, corporates and other undesirables the ability to put a gun to the head of the messenger and legally demand – like a cat burying its feces in the ground – that all trace of embarrassment should be removed. Like cat shit, this stinks.

Of course, like any publisher or voice of authority, there needs to be a balance. And that balance needs to be firmly placed with those who write and publish on the Internet. Never has the need for ethics, oversight and just sheer common sense been more required than in these days of Facebook, Twitter and the Blog. The Duke of Wellington said "Publish and be damned", but that was more akin to the Gaelic shrug of indifference than

to the current understanding – that publishers must take responsibility for what is printed (albeit in electronic format) – is a good ethical principle.

Forcing Google to remove links is akin to breaking into a library and removing the Dewey index card for whatever book offends the reader in a library. The content is still there, but hey ho, situation ethics wins so my only hope is that the Streisand effect will kick in and give those that choose to go down this route a run for their money.

Let's be clear though, a lie goes halfway round the world before the truth has got its boots on. Everyone – Publishers, Google and ISP's have a responsibility to ensure that the Internet is a truthful and valid source of information. And here we really start to open a very large can of worms – for throughout the ages the powers that be (or the aliens, take your pick) have sought to corrupt, debase, and change the very fine thread of truth that characterizes a particular scenario. Propaganda, spin, advertising, statistics – all of these have a huge impact on human consciousness. And the Internet is rife with them all. Again it goes back to the publishers, but maybe we need some discipline on the the good old Interweb. Where do we draw the line between entertainment, opinion, experience, knowledge and truth?

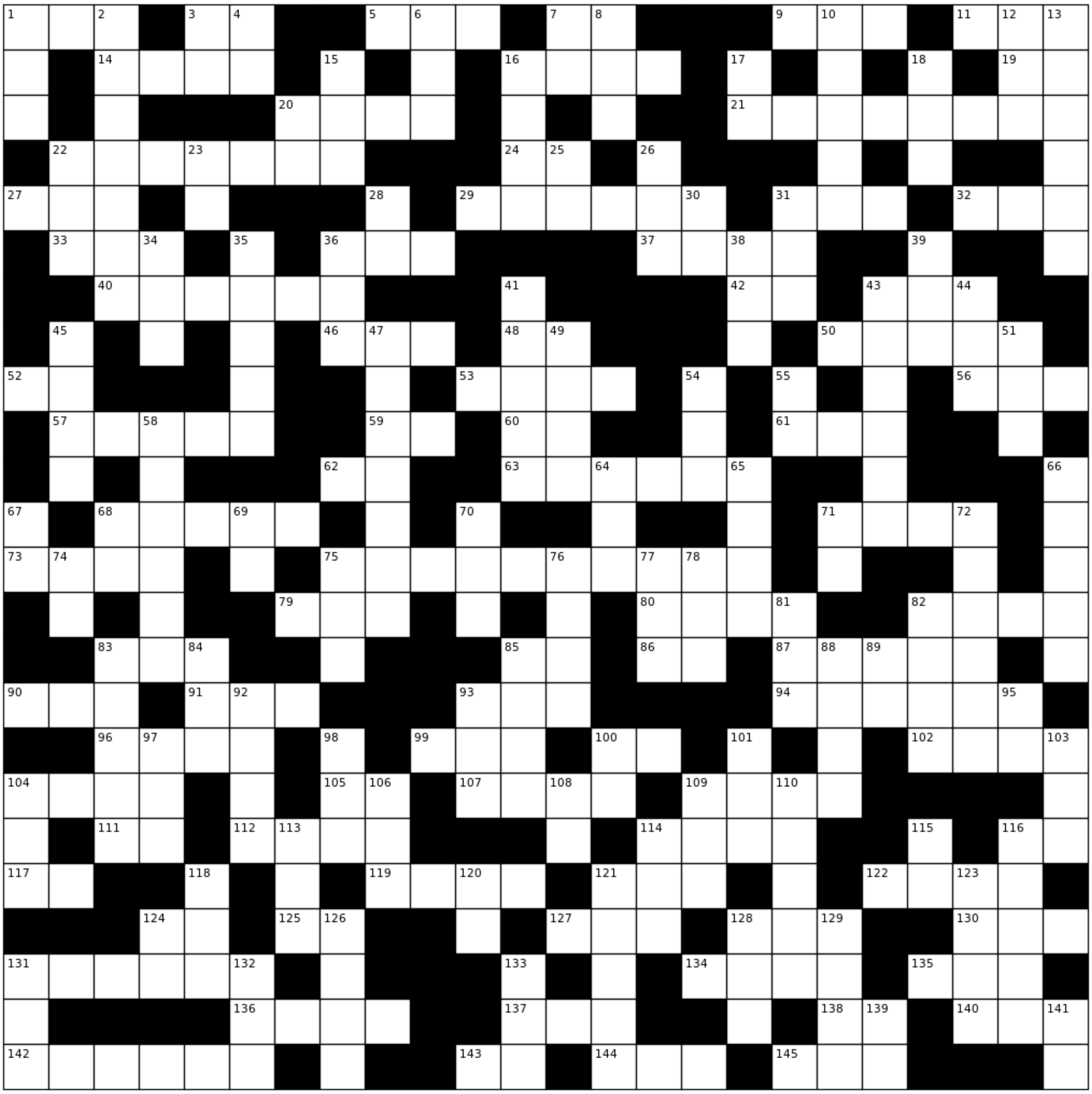
I'd love to be able to come up with a definitive answer, but like the human condition, the answer is a bit more complex. Google, like the Internet, is just a mirror placed at the face of humanity. Those that choose to scrape the silver coating off the back so that a true reflection will not be seen will so crudely distort the image that the cries of "foul" will be louder than their misdeeds, and thereby attract more attention to themselves. That is, of course, if Google has the courage to display the link with the words "We have been asked to remove this link because ….." Otherwise, we will have an interesting scenario whereby Google can comply with the Digital Millennium Copyright Act by leaving a link while erasing more interesting material without a trace.

I really hope Google is as enraged about this as they are pretending to be. Bowing down to the DMCA was bad enough but to crumple on this will take "Do no evil" to the new low of "Do nothing".

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.

The FreeBSD Cryptic Crossword



Across

1 Stock exchange abbreviation for Bill Gates' empire (3)
3 After midday (2)
5 Linux desktop here be dragons! (3)
7 (and 7 down) In programming, either (2)
9 HTML page request (3)
11 Abbreviation for menu that appears on monitor, independent of operating system (3)
14 Microsoft document format now supported by Open Office (4)
16 Unix utility to search files for content (4)
19 Decimal 202 in hexadecimal (2)
20 Linux sound module (4)
21 A physical computer, service or device (8)
22 VMware and Xen run these types of devices – a science fiction reality? (7)
24 Motherboard manufacturer that has 1024 Kilobytes? (2)
27 Sound file suffix (3)
29 Very loud data drive or large book? (6)
31 Abbreviation for interrupt or integer (3)
32 Commercial brand of Unix that tried to sue but lost (3)
33 Color of the FreeBSD daemon (3)
36 Microsoft outlook mail file suffix (3)
37 Initialise a random number or encryption command plant kernel? (4)
40 Hangup signal (6)
42 Truncated executable suffix? (2)
43 Sum two numbers or abbreviation for the medical condition where a person cannot pay attention (3)
46 BSD software comes in this format (3)
48 Switch used with ls to generate output separated with commas (2)
50 Trendy hardware and software manufacturer that has eaten fruit (5)
52 Abbreviation for readonly (2)
53 Single or multiplayer 80's game that is quite negative (4)
56 Abbreviation for line printer (3)
57 Mature programming language used on mainframes and sometimes PC's might be cool? (5)
59 Unix equivalent of DIR (2)
60 Part of a DOS new line abbreviation for horse drawn vehicle going back? (2)
61 Abbreviation for a network card (3)
62 Abbreviation for input and output (2)
63 Core of Linux and *BSD's You do not want this to 83 down (6)
68 Copy MSDOS files to/from Unix (5)
71 Generally the next operation after opening a file or book (4)
73 Elderly database engine that used
variable length fields (4)
75 UFS, JFS, and EXT are all this (10)
79 American telco closely allied to Unix (3)
80 You cannot append to a file unless you do this first (4)
82 Not zero and not empty (4)
83 (and 36 down) Mature communications protocol 3 green vegetables in a pod? (3)
85 Abbreviation for Micro$oft (2)
86 Unix utility to list processes or a printer language (2)
87 Protocol essential to the Internet (5)

Across

90 US government agency that reverses encryption (3)
91 IBM's PC bus (3)
93 Hardware that can be swapped when running – CPU gets this? (3)
94 Programming language named after a mathematician? (6)
96 Point on a data tree – content provided by 72 down? (4)
99 Compressed file type extension – not related to clothes? (3)
100 Abbreviation for department that has sys admins and programmers (2)
102 Next programming condition after if then .... (4)
104 Unix folder that holds supervisor binaries (4)
105 Early text editor – not 115 down (2)
107 Detachable power cable – or heavy metal? (4)
109 HTML page request – sent in the mail? (4)
111 Decimal 206 in Hexadecimal (2)
112 1000 or 1024 depending on the definition (4)
114 Common Linux version of 35 down (4)
116 Decimal 255 in Hexadecimal (2)
117 Unix utility to show disk space used (2)
119 To relocate a file from one place to another (4)
121 Extended version of 115 down abrasive powder cleaner? (3)
122 Rudely terminate a process (4)
124 Small hand held games console (2)
125 This executable would run the turbo version of 94 across on a Microsoft box – Indian wigwam? (2)
127 Opposite of XOFF handshake (3)
128 Very old hard drive architecture would not compete in a marathon? (3)
130 German enterprise software – or the interior of a moist plant? (3)
131 To remove a file, but not always permanently (6)
134 A female connector is useless without this (4)
135 Audit of a process written to file (3)
136 Unix GUI display committee? (4)
137 A common firewall or router function an annoying insect maybe? (3)
138 Type of memory card (2)
140 PC bus – not only found in Russian secret service? (3)
142 Process that had died but not closed down neatly (6)
143 In Unix, change to another user – not always root (2)
144 (and 104 down) Unix editor that lives in a river? (3)
145 PC bus that provides power everywhere? (3)

Down

1 Module for 43 down – or a forum manager? (3)
2 Multiple bootable hard disks for DOS or Windows O/S (7)
3 The IBM XT is this, a Silicon Graphics workstation is not (2)
4 Essential DNS record for email (2)
6 Method of accessing RAM – to put it bluntly (3)
7 (and 7 across) In programming, either (2)
8 Comment out a program line in BASIC (3)
10 Could be down to faulty software or hardware (5)
12 Microsoft screen saver file suffix – first three letters of modern display device (3)
13 Unix process that runs in the background (6)
15 (and 44 down) Microsoft library (3)

Down

16 Acronym for "If you put rubbish in you get rubbish" (4)
17 Tell a script to run under 35 down #!/bin/ often precedes it (2)
18 Spider casts its trap across the earth but a domain doesn't always need this (3)
20 DNS record for a web page (2)
22 Where 135 across is often stored (3)
23 Abbreviation for amount of traffic sent via 61 across (2)
25 Hexadecimal 42 4C or decimal 066 076 in ASCII (2)
26 Operating system common to DEC (3)
28 Abbreviation for Windows, Unix and CP/M (2)
30 Decimal 238 in Hexadecimal (2)
31 Modern version of 128 across – or a programmers workbench? (3)
34 Unix utility to query DNS (3)
35 (and 110 down) TCSH, CSH are this but only found near the ocean (5)
36 (and 83 across) Mature communications protocol – 3 green vegetables in a pod? (3)
38 Microsoft binary executable and a well known programmers magazine (3)
39 Insecure protocol for accessing desktops (3)
41 Basic measurement of hard disk or tape storage (5)
43 Well known web server software – Indian foundation? (6)
44 (and 15 across) Microsoft library (3)
45 A file or database will create this to prevent overwriting (4)
47 One thousand bits (7)
49 Opposite of 129 down (4)
51 (and 78 down) Encapsulated version of 89 down (3)
54 Perl or PHP error handling statement a metal former perhaps? (3)
55 Binary 1, set or enabled (2)
58 Make copy of files to prevent disaster (6)
64 In Linux, variables that affect the behaviour of boot scripts (3)
65 Unix software to convert audio files (4)
66 Hardware or software process continually checking – always happens during political elections (5)
67 Hardware manufacturer that started in a garage (2)
68 Unix file manager – an advanced version of 110 down (2)
69 Printer language or Unix command to list processes (2)
70 Bright diode found in most hardware (3)
71 Abbreviation that says you cannot write (2)
72 Popular open source Content Management System (6)
74 Abbreviation for Twitter or a chat server conversation – quick chat? (2)
75 Insecure transfer protocol (3)
76 SUSE Linux management software (4)
77 More colourful version of 69 down? (3)
78 (and 51 down) Encapsulated version of 89 down (3)
81 Server that distributes time on a network (3)
82 Unix utility to raise or lower a process priority (4)
83 You may do this if 63 across fails to load or run correctly (5)
84 The number of a process – a project manager's first document? (3)
85 File perusal filter for crt viewing (4)
88 To change a number type in C – a support for a broken limb? (4)
89 Printer and graphics language (2)
92 A drive head will do this (4)
93 Linux abstraction layer for 21 across (3)

Down

95 Unix command equivalent to DOS DIR (2)
97 Binary 1 in single word (3)
98 Two of these keys will be found on your keyboard – related to 131 across (3)
100 SCSI device number (2)
101 Microsoft's text only O/S (3)
103 A common Linux binary format mythical small creature? (3)
104 (and 144 across) Unix editor that lives in a river? (3)
106 Object orientated model for web developers – or a short XEN domain ? (3)
108 Decimal 174 in Hexadecimal (2)
109 Unix authentication module or arbitrary map file format – her name has been shortened? (3)
110 (and 35 down) TCSH, CSH are this but only found near the ocean (5)
113 Abbreviation for interrupt (3)
114 Unix binaries are found in this folder (3)
115 Alternative editor to Emacs (2)
116 Compiler switch – country identifier by fabric? (5)
118 Abbreviation for an American time in the summer (3)
120 Shorthand for 22 across if it is a server or desktop (2)
121 Current times resistance – the answer is shocking (5)
123 In linux, list open files (4)
124 Decimal 222 in Hexadecimal (2)
126 Remote or local access point on a server – every sailor visits one (4)
128 Memory, accessed in a nonlinear fashion (3)
129 Opposite of 85 down (4)
131 Firewall area – In a war this is the place to be (3)
132 Microsoft suffix for an executable (3)
133 This is not Unix (3)
139 Abbreviation of Dr Codd's system (2)
141 HTML new line tag (2)

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.

Great Specials

On FreeBSD® & PC-BSD® Merchandise

Give us a call & ask about our SOFTWARE BUNDLES
1.925.240.6652

$39.95 FreeBSD 9.1 Jewel Case CD Set or FreeBSD 9.1 DVD

$29.95 PC-BSD 9.1 DVD

$49.95 Save a BUNDLE! The PC-BSD 9.0 Users Handbook PC-BSD 9.1 DVD

$99.95 The FreeBSD CD or DVD Bundle

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

Stylish Dress Attire
Look Your Professional Best
Comfy Apparel
Stay Warm in Zip Ups & Pullovers
T-Shirts
Lots of Styles to Choose From

FreeBSD 9.1 Jewel Case CD/DVD $39.95

CD Set Contains:

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

FreeBSD 9.0 CD $39.95
FreeBSD 9.0 DVD $39.95

FreeBSD Subscriptions

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

FreeBSD Subscription, start with CD 9.1 .... $29.95
FreeBSD Subscription, start with DVD 9.1 .... $29.95
FreeBSD Subscription, start with CD 9.0 .... $29.95
FreeBSD Subscription, start with DVD 9.0 .... $29.95

PC-BSD 9.1 DVD (Isotope Edition)

PC-BSD 9.1 DVD $29.95
PC-BSD Subscription $19.95

The FreeBSD Handbook

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

The FreeBSD Handbook Specials

The FreeBSD Handbook, Volume 2 (Both Volumes) $59.95
The FreeBSD Handbook, Both Volumes & FreeBSD 9.1 .... $79.95

PC-BSD 9.0 Users Handbook $24.95

BSD Magazine $11.99

The FreeBSD Toolkit DVD $39.95

FreeBSD Mousepad $10.00

FreeBSD & PCBSD Caps $20.00

BSD Daemon Horns $2.00

Bundle Specials!
Save $$$

Just Plain Fun
Mousepads & Novelty Horns

BSD Magazine
Available Monthly

FreeBSD Mall

For even MORE items visit our website today!
www.FreeBSDMall.com

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

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

OUR EXPERTISE INCLUDES:

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

Net Open SERVICES

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

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

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

WWW.NETOPENSERVICES.COM • CONTACT@NETOPENSERVICES.COM