BSD MAGAZINE

FOR NOVICE AND ADVANCED USERS

BSD -CURRENT is Usable Daily

FREEBSD / OPENBSD / DRAGONFLY SYSTEMS

SIMULATIONS AND SECURITY PROCESSES

GDB DEBUGGER

INSTALLING THE E-MAIL SERVERS AND THE WEBMAIL INTERFACE

VOL.9 NO.02
ISSUE 67
1898-9144

iXsystems®

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

✓ Rock-Solid Performance

✓ Professional In-House Support

FREENAS MINI
STORAGE APPLIANCE

IT *SAVES* YOUR LIFE.

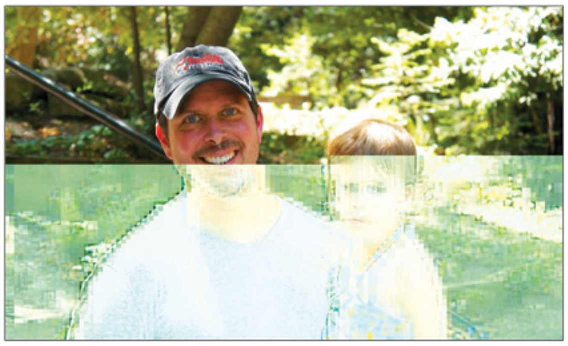
HOW IMPORTANT IS YOUR DATA?

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®

http://www.iXsystems.com/mini

intel inside™ ATOM™

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...

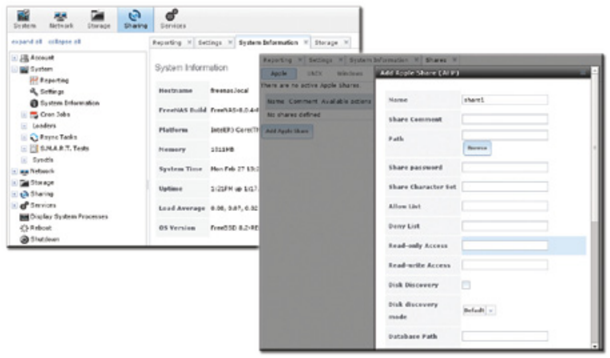
MAKE IT EASY ON YOURSELF

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 15:
Uptime	3:21PM up 1:17
Load Average	0.00, 0.07, 0.02
OS Version	FreeBSD 8.2-RE

There are no active Apple shares.

No shares defined

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,

Nice to meet you again. Now you are going to read the next issue from BSD magazine. You have the chance to walk through the installation and the basic configuration of Postfix, one of the most popular SMTP servers, and SpamAssassin, which will be used for basic e-mail filtering.

What is more, our experts will show you that most of the companies stick to stable/release versions with only security fixes. Indeed, if your applications rely on specific API/ABI versions, it is better to keep on doing it, but others run experimental branches. You will learn more from our article written by David Carlier.

Finally, you may find interest in the article from the Technologies section. The article gives you more insight into industry practices.

We would like to express our gratitude to our experts who contributed to this publication and invite others to cooperate with our magazine.

Enjoy reading,
Ewa & the 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

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.

FreeNAS
in an Enterprise Environment

By the time you're reading this, FreeNAS has been downloaded more than 5.5 million times. For home users, it's become an indispensable part of their daily lives, akin to the DVR. Meanwhile, all over the world, thousands of businesses universities, and government departments use FreeNAS to build effective storage solutions in myriad applications.

What you will learn...

- How TrueNAS builds off the strong points of the FreeBSD and FreeNAS operating systems
- How TrueNAS meets modern storage challenges for enterp

The FreeNAS operating systems is fre
the public and offers thorough doc
active community, and a feature-ri
the storage environment. Based on Free
can share over a host of protocols (SMB
FTP, iSCSI, etc) and features an intuitiv
the ZFS file system, a plug-in system
much more.

Despite the massive popularity
aren't aware of its big brother duti
data in some of the most demand
environments: the proven, enterp
professionally-supported line of

But what makes TrueNAS diffe
Well, I'm glad you asked...

Commercial Grade Supp

When a mission critical stor
organization's whole operat
halt. Whole community-bas
free), it can't always get an
and running in a timely m
responsiveness and expe
dedicated support team
provide that safety.

Created by the sam
developed FreeNAS.

NEW RELEASE

iX systems®

WE INTERRUPT THIS MAGAZINE TO BRING YOU THIS IMPORTANT ANNOUNCEMENT:

THE PEOPLE WHO DEVELOP FREENAS, THE WORLD'S MOST POPULAR STORAGE OS, HAVE JUST REVAMPED TRUENAS.

**POWER WITHOUT CONTROL MEANS NOTHING.
TRUENAS STORAGE GIVES YOU BOTH.**

- ☑ Simple Management
- ☑ Hybrid Flash Acceleration
- ☑ Intelligent Compresssion
- ☑ All Features Provided Up Front (no hidden licensing fees)
- ☑ Self-Healing Filesystem
- ☑ High Availability
- ☑ Qualified for VMware and HyperV
- ☑ Works Great With Citrix XenServer®

To learn more, visit: www.iXsystems.com/truenas

intel® Xeon® inside™

POWERED BY INTEL® XEON® PROCESSORS

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

VMware and VMware Ready are registered trademarks or trademarks of VMware, Inc. in the United States and other jurisdictions.

Citrix makes and you receive no representations or warranties of any kind with respect to the third party products, its functionality, the test(s) or the results there from, whether expressed, implied, statutory or otherwise, including without limitation those of fitness for a particular purpose, merchantability, non-infringement or title. To the extent permitted by applicable law. In no event shall Citrix be liable for any damages of any kind whatsoever arising out of your use of the third party product, whether direct, indirect, special, consequential, incidental, multiple, punitive or other damages.

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.

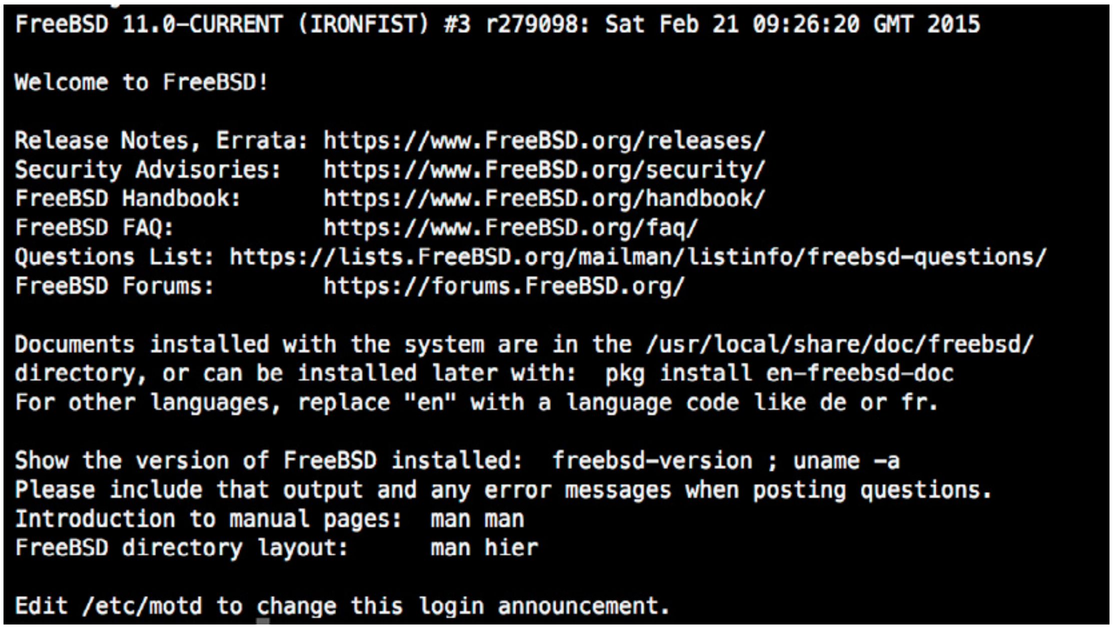
WWW.NETOPENSERVICES.COM • CONTACT@NETOPENSERVICES.COM

BSD -CURRENT is Usable Daily

DAVID CARLIER

Running the development branch of a *BSD daily might sound scary. Indeed, this is basically the experimentations' land and this use case seems to apply only to BSD developers – the internal APIs might suddenly change because they need to, some bugs can be fixed, some new ones can be introduced without notice (although in general, the community is quite reactive and fixes them fairly quickly). I am going to talk about the BSDs I know and use the most and I'll explain the reasons for using the -CURRENT branches.

One of the main reasons which I use -CURRENT branches is simply having the latest innovations. In the case of FreeBSD, I like having the very latest version possible of clang because I am following the coming of some expected features, like OpenMP support and sanitizers support, because of the compilation effectiveness improvements, and so on. As I often use virtualized environments, having the latest bhyve features is a very good point. From a developer point of view, having new syscalls like explicit_bzero (which can be preferred in place of memset for some use cases, avoiding the potential compiler optimization ...), ppoll for the Linux emulation layer are beneficial. Casperd pro-



```
FreeBSD 11.0-CURRENT (IRONFIST) #3 r279098: Sat Feb 21 09:26:20 GMT 2015

Welcome to FreeBSD!

Release Notes, Errata: https://www.FreeBSD.org/releases/
Security Advisories:   https://www.FreeBSD.org/security/
FreeBSD Handbook:      https://www.FreeBSD.org/handbook/
FreeBSD FAQ:           https://www.FreeBSD.org/faq/
Questions List: https://lists.FreeBSD.org/mailman/listinfo/freebsd-questions/
FreeBSD Forums:        https://forums.FreeBSD.org/

Documents installed with the system are in the /usr/local/share/doc/freebsd/
directory, or can be installed later with:  pkg install en-freebsd-doc
For other languages, replace "en" with a language code like de or fr.

Show the version of FreeBSD installed:  freebsd-version ; uname -a
Please include that output and any error messages when posting questions.
Introduction to manual pages:  man man
FreeBSD directory layout:      man hier

Edit /etc/motd to change this login announcement.
```

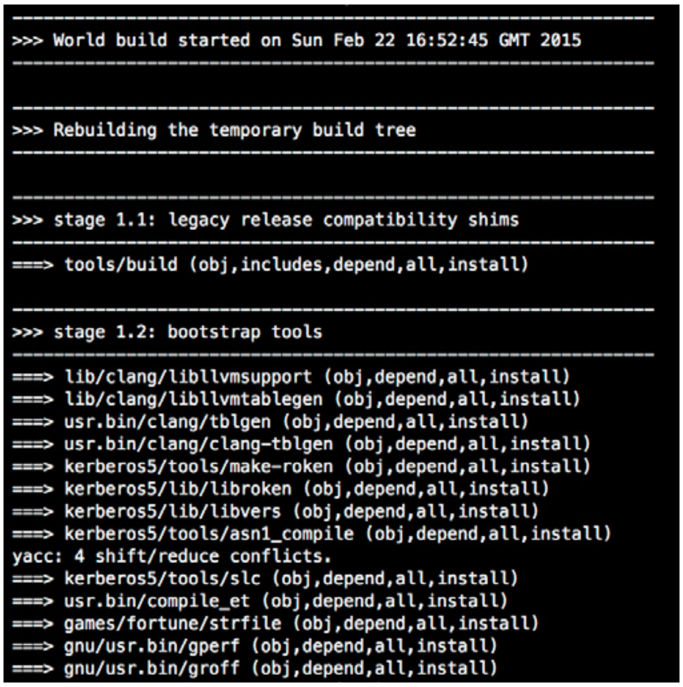
Figure 1. *FreeBSD CURRENT*

vides some services not available in capsicum's capabilities mode and can be seen as a proxy, for example, for DNS resolution.

For OpenBSD, having the latest relayd/httpd features interests me (i.e., I run a custom version of relayd which produces some additional custom HTTP headers). I appreciate their "backward compatibility breaking fearless for the better good" approach (the recent change in random C API, for example, could confirm it). Indeed since the 5.6, the static Position Independent Executable support for base system binaries was added, the legacy deterministic rand C API was strongly updated, and so on ...

I recently retried NetBSD, with LLVM/clang in base following their willingness to move towards it. After some days of usage, I noticed a general small performance drop (one of my custom applications got something like 5/10 percent of difference) but it is a generally well known problem with clang; it is improving through the releases.

Lastly, DragonflyBSD recently brought GCC 5.0 in base (with a bunch of new sanitizations flags, in addition to the OpenMP 4.0 specifications support). Also more generally, a lot of effort is made in the graphic stack. Having the last fixes for Hammer1 filesystem is worthwhile (i.e Hammer2 is still not production ready).



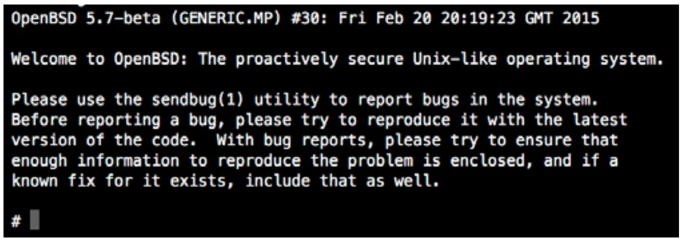
```
--------------------------------------------------------------
>>> World build started on Sun Feb 22 16:52:45 GMT 2015
--------------------------------------------------------------

--------------------------------------------------------------
>>> Rebuilding the temporary build tree
--------------------------------------------------------------

--------------------------------------------------------------
>>> stage 1.1: legacy release compatibility shims
--------------------------------------------------------------
===> tools/build (obj,includes,depend,all,install)

--------------------------------------------------------------
>>> stage 1.2: bootstrap tools
--------------------------------------------------------------
===> lib/clang/libllvmsupport (obj,depend,all,install)
===> lib/clang/libllvmtablegen (obj,depend,all,install)
===> usr.bin/clang/tblgen (obj,depend,all,install)
===> usr.bin/clang/clang-tblgen (obj,depend,all,install)
===> kerberos5/tools/make-roken (obj,depend,all,install)
===> kerberos5/lib/libroken (obj,depend,all,install)
===> kerberos5/lib/libvers (obj,depend,all,install)
===> kerberos5/tools/asn1_compile (obj,depend,all,install)
yacc: 4 shift/reduce conflicts.
===> kerberos5/tools/slc (obj,depend,all,install)
===> usr.bin/compile_et (obj,depend,all,install)
===> games/fortune/strfile (obj,depend,all,install)
===> gnu/usr.bin/gperf (obj,depend,all,install)
===> gnu/usr.bin/groff (obj,depend,all,install)
```

Figure 2. *Recompiling FreeBSD ... The time needed is fairly variable*

One of the downsides of running current is if you're using a desktop environment or more generally the ports system. In general, when a significant change in the base system occurs, it is recommended to rebuild all the ports afterwards. The time needed to do so could be potentially quite important, especially with software like KDE, Gnome 3, etc. It is a point to weigh well ...

For FreeBSD -CURRENT, I very rarely run a desktop. I prefer to use the whole potential CPU/memory for compiling the system instead. Also, the fact that I enable a significant amount of debugging kernel options which slow down the general performance (like WITNESS (to detect potential deadlocks) / INVARIANTS (which add more kernel level's assertion) flags) stops me from considering it. Those specific options are only useful for developers or beta testers though. It is advised to disable them otherwise.

In the case of OpenBSD -CURRENT, I run time in time the base cwn which is very light and xorg (called xenocara) is not in the ports but in the base system, that makes those updates easier. In addition, I enable MALLOC_STATS, hence allowing the D flag for MALLOC_OPTIONS for debugging purposes with the cost at a performance hit. Again, this last one is not recommended if you are not a developer.



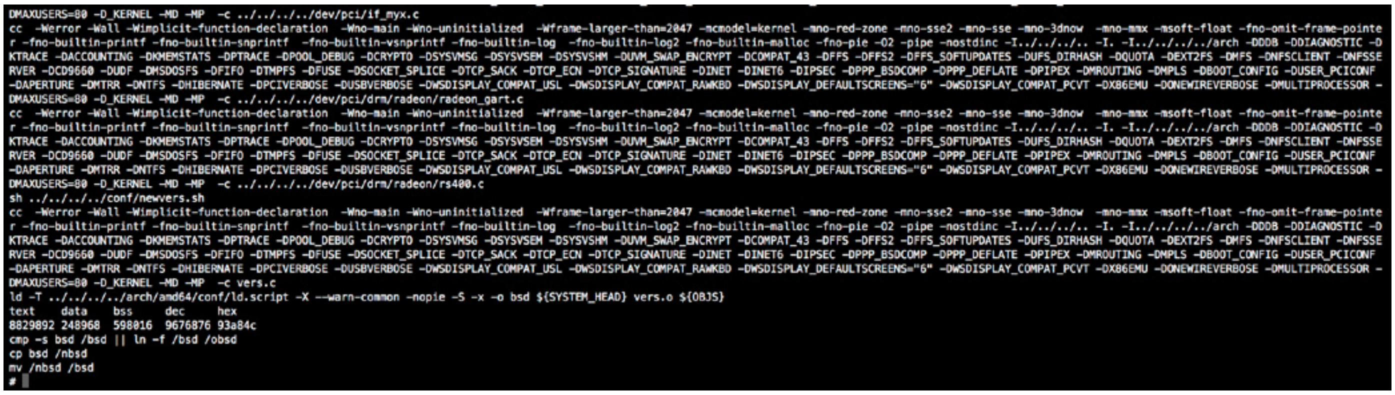
```
OpenBSD 5.7-beta (GENERIC.MP) #30: Fri Feb 20 20:19:23 GMT 2015

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

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

# █
```

Figure 3. *OpenBSD 5.7-BETA, close to the next release*

From a company point of view, if a new feature is genuinely needed and if it is not possible to do it internally, sponsorship might be considered an option.

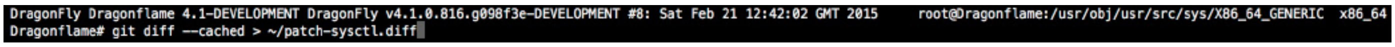
Bug acceptability level

Indeed, the -CURRENT branches introduce potentially some new bugs. In the case of FreeBSD, for example, recently the Random Number Generator framework change, which was made pluggable, was found to be broken. Instead of coming back to the previous version, which sounds less risky, the issue was fixed – I personally prefer this kind of approach. On my side, I run FreeBSD with some local fixes (for bsdgrep, for example), some were merged upstream, hopefully some others will be in the near future.



```
DMAXUSERS=80 -D_KERNEL -MD -MP  -c ../../../../dev/pci/if_myx.c
cc  -Werror -Wall -Wimplicit-function-declaration  -Wno-main -Wno-uninitialized  -Wframe-larger-than=2047 -mcmodel=kernel -mno-red-zone -mno-sse2 -mno-sse -mno-3dnow  -mno-mmx -msoft-float -fno-omit-frame-pointe
r -fno-builtin-printf -fno-builtin-snprintf  -fno-builtin-vsnprintf -fno-builtin-log  -fno-builtin-log2 -fno-builtin-malloc -fno-pie -O2 -pipe -nostdinc -I../../../.. -I. -I../../../../arch -DDDB -DDIAGNOSTIC -D
KTRACE -DACCOUNTING -DKMEMSTATS -DPTRACE -DPOOL_DEBUG -DCRYPTO -DSYSVMSG -DSYSVSEM -DSYSVSHM -DUVM_SWAP_ENCRYPT -DCOMPAT_43 -DFFS -DFFS2 -DFFS_SOFTUPDATES -DUFS_DIRHASH -DQUOTA -DEXT2FS -DMFS -DNFSCLIENT -DNFSSE
RVER -DCD9660 -DUDF -DMSDOSFS -DFIFO -DTMPFS -DFUSE -DSOCKET_SPLICE -DTCP_SACK -DTCP_ECN -DTCP_SIGNATURE -DINET -DINET6 -DIPSEC -DPPP_BSDCOMP -DPPP_DEFLATE -DPIPEX -DMROUTING -DMPLS -DBOOT_CONFIG -DUSER_PCICONF
-DAPERTURE -DMTRR -DNTFS -DHIBERNATE -DPCIVERBOSE -DUSBVERBOSE -DWSDISPLAY_COMPAT_USL -DWSDISPLAY_COMPAT_RAWKBD -DWSDISPLAY_DEFAULTSCREENS="6" -DWSDISPLAY_COMPAT_PCVT -DX86EMU -DONEWIREVERBOSE -DMULTIPROCESSOR -
DMAXUSERS=80 -D_KERNEL -MD -MP  -c ../../../../dev/pci/drm/radeon/radeon_gart.c
cc  -Werror -Wall -Wimplicit-function-declaration  -Wno-main -Wno-uninitialized  -Wframe-larger-than=2047 -mcmodel=kernel -mno-red-zone -mno-sse2 -mno-sse -mno-3dnow  -mno-mmx -msoft-float -fno-omit-frame-pointe
r -fno-builtin-printf -fno-builtin-snprintf  -fno-builtin-vsnprintf -fno-builtin-log  -fno-builtin-log2 -fno-builtin-malloc -fno-pie -O2 -pipe -nostdinc -I../../../.. -I. -I../../../../arch -DDDB -DDIAGNOSTIC -D
KTRACE -DACCOUNTING -DKMEMSTATS -DPTRACE -DPOOL_DEBUG -DCRYPTO -DSYSVMSG -DSYSVSEM -DSYSVSHM -DUVM_SWAP_ENCRYPT -DCOMPAT_43 -DFFS -DFFS2 -DFFS_SOFTUPDATES -DUFS_DIRHASH -DQUOTA -DEXT2FS -DMFS -DNFSCLIENT -DNFSSE
RVER -DCD9660 -DUDF -DMSDOSFS -DFIFO -DTMPFS -DFUSE -DSOCKET_SPLICE -DTCP_SACK -DTCP_ECN -DTCP_SIGNATURE -DINET -DINET6 -DIPSEC -DPPP_BSDCOMP -DPPP_DEFLATE -DPIPEX -DMROUTING -DMPLS -DBOOT_CONFIG -DUSER_PCICONF
-DAPERTURE -DMTRR -DNTFS -DHIBERNATE -DPCIVERBOSE -DUSBVERBOSE -DWSDISPLAY_COMPAT_USL -DWSDISPLAY_COMPAT_RAWKBD -DWSDISPLAY_DEFAULTSCREENS="6" -DWSDISPLAY_COMPAT_PCVT -DX86EMU -DONEWIREVERBOSE -DMULTIPROCESSOR -
DMAXUSERS=80 -D_KERNEL -MD -MP  -c ../../../../dev/pci/drm/radeon/rs400.c
sh ../../../../conf/newvers.sh
cc  -Werror -Wall -Wimplicit-function-declaration  -Wno-main -Wno-uninitialized  -Wframe-larger-than=2047 -mcmodel=kernel -mno-red-zone -mno-sse2 -mno-sse -mno-3dnow  -mno-mmx -msoft-float -fno-omit-frame-pointe
r -fno-builtin-printf -fno-builtin-snprintf  -fno-builtin-vsnprintf -fno-builtin-log  -fno-builtin-log2 -fno-builtin-malloc -fno-pie -O2 -pipe -nostdinc -I../../../.. -I. -I../../../../arch -DDDB -DDIAGNOSTIC -D
KTRACE -DACCOUNTING -DKMEMSTATS -DPTRACE -DPOOL_DEBUG -DCRYPTO -DSYSVMSG -DSYSVSEM -DSYSVSHM -DUVM_SWAP_ENCRYPT -DCOMPAT_43 -DFFS -DFFS2 -DFFS_SOFTUPDATES -DUFS_DIRHASH -DQUOTA -DEXT2FS -DMFS -DNFSCLIENT -DNFSSE
RVER -DCD9660 -DUDF -DMSDOSFS -DFIFO -DTMPFS -DFUSE -DSOCKET_SPLICE -DTCP_SACK -DTCP_ECN -DTCP_SIGNATURE -DINET -DINET6 -DIPSEC -DPPP_BSDCOMP -DPPP_DEFLATE -DPIPEX -DMROUTING -DMPLS -DBOOT_CONFIG -DUSER_PCICONF
-DAPERTURE -DMTRR -DNTFS -DHIBERNATE -DPCIVERBOSE -DUSBVERBOSE -DWSDISPLAY_COMPAT_USL -DWSDISPLAY_COMPAT_RAWKBD -DWSDISPLAY_DEFAULTSCREENS="6" -DWSDISPLAY_COMPAT_PCVT -DX86EMU -DONEWIREVERBOSE -DMULTIPROCESSOR -
DMAXUSERS=80 -D_KERNEL -MD -MP  -c vers.c
ld -T ../../../../arch/amd64/conf/ld.script -X --warn-common -nopie -S -x -o bsd ${SYSTEM_HEAD} vers.o ${OBJS}
text    data    bss     dec     hex
8829892 248968  598016  9676876 93a84c
cmp -s bsd /bsd || ln -f /bsd /obsd
cp bsd /nbsd
mv /nbsd /bsd
# 
```

Figure 4. *Recompiling OpenBSD is a quite simple task*



```
DragonFly Dragonflame 4.1-DEVELOPMENT DragonFly v4.1.0.816.g098f3e-DEVELOPMENT #8: Sat Feb 21 12:42:02 GMT 2015     root@Dragonflame:/usr/obj/usr/src/sys/X86_64_GENERIC  x86_64
Dragonflame# git diff --cached > ~/patch-sysctl.diff
```

Figure 5. *DragonflyBSD uses git, better for branches handling*

In the case of OpenBSD, the new XHCI driver (for USB 3.0) still does not work completely. For example, recently a memory leak was found in dhclient (but fixed) ... But nothing really major, OpenBSD -CURRENT is runnable daily as well.

DragonflyBSD had memory leaks in the kernel and in hammer filesystem. Once again, they were fixed promptly.

The bug "acceptability" level depends on whether you're willing to patiently take the time to make explicit bug reports in case the bug in question is blocking, or fixing them internally and pushing those fixes upstream. But there is no support to expect – again a point to consider well.

Contribution

Most of the contributions are done in the -CURRENT branches. That makes perfect sense as the -CURRENT branches are the perfect areas for both fixes and innovative features, adding disruptive changes whereas the releases/stables welcome the fixes only. It also makes more sense for -CURRENT that recompiling the system is the natural usage.

If you are a quite advanced BSD user and you wish to contribute to make them better for the whole community then using the development branches can be considered. There are many areas, not only purely technical (like the documentation) which can be improved.

DragonflyBSD uses git internally and due to its branching model, it is pretty handy to create a proper diff to submit it for review.

Conclusion

Most companies stick to stable/release versions with only security fixes. If your applications rely on specific API/ABI versions, it is indeed better to keep on doing it.

Somehow, few others run experimental branches. Indeed. For example, Yahoo uses FreeBSD -CURRENT internally for their servers.

Given the short life release cycle chosen by OpenBSD with its fair amount of disruptive changes (ie., every 6 months), it is less surprising to find users using the development branch.

I recompile quite often FreeBSD / OpenBSD base systems but for those who have no interest at all in doing it, some snapshots builds are made fairly often ...

Saying that, it is advised to be registered in the relevant mailing lists: *freebsd-current@freebsd.org, tech@openbsd.org, tech@netbsd.org, commits@dragonflybsd.org.*

ABOUT THE AUTHOR

David Carlier has been working as a software developer since 2001. He used FreeBSD for more than 10 years and starting from this year, he became involved with the HardenedBSD project and performed serious developments on FreeBSD. He worked for a mobile product company that provides C++ APIs for two years in Ireland. From this, he became completely inspired to develop on FreeBSD.

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

Installing the E-mail Servers and the Webmail Interface

IVAN VORAS

The goal of this tutorial is to walk readers through the installation and the basic configuration of Postfix, one of the most popular SMTP servers, and SpamAssassin, which will be used for basic e-mail filtering.

SMTP is the protocol used to route e-mail messages to and between servers. The delivery of e-mail to user-facing software, such as e-mail clients like Thunderbird, Outlook and others, is the job of other protocols like IMAP (and the old, obsolete POP3).

SMTP was made for a smaller and more trustworthy Internet and offers next to no guarantees that a message was sent from a valid user, to a valid user, or that it will arrive in time. However, it offers decent micro-guarantees about what happens if a message is received by a server. In particular, it offers store-and-forward semantics in which the SMTP server receiving a message promises that, if it acknowledges that the message was received, it has successfully stored the message and will do its best to forward it to its intended recipient. The first feature makes sending unrequested and forged e-mail very easy (we call it *spam*), and the second feature makes processing e-mail fairly resource intensive, as it involved synchronous writing of the messages on the server's drives. Because of this, running e-mail servers is harder than it should be. Today, SMTP servers are heavily guarded by firewalls, antiviruses and strict rules about who is permitted to send e-mail through them.

SMTP and DNS

A modern e-mail message has two parts: the username and the domain part. We will cover the username part later, but the domain part needs some special consideration. As when accessing a web site, an application needs to use the DNS system to find out the IP address of the server which is accessed. To allow e-mail to be served by different servers than those which serve other services for the domain, a special type of DNS entry is necessary, called the "MX record" (stands for Mail eXchanger). When sending e-mail to an address such as "user@example.com", first the MX record for "example.com" is searched. If it's not found, a regular "A" record is searched.

Note that DNS must be resolvable for e-mail to work, as is true with other services, such as the web.

In addition to MX records, the DNS system can also carry SPF records (Sender Policy Framework) which can be used to inform the world about which IP addresses are allowed to send e-mail on behalf of which domains. This can be used to reduce the possibility of forged e-mails for certain domains.

Complex e-mail routing

In some cases, the system which receives e-mail for a domain is not the final server which will store the e-mail in the users' mailboxes. These cases require that e-mail be routed from a server to a server, usually from a more general server (such as a global organization's server) to a more specific server (such as a local branch's server). In such a scenario, it is possible for the servers which are internal to the organization to have local IP addresses, though it requires a careful design of the network and its services.

E-mail usernames

On Unix-like systems, the usernames used in e-mail addresses are usually the system usernames. In such systems, e-mail is delivered to locations provided by the system, such as the users' home directories, and are protected by the system access protection rules (such as file ownership and access permissions).

This is not necessarily so: e-mail usernames could be stored in a database and delivered to special mailboxes, but such setups are outside the scope of this article.

Spam protection

Since incoming e-mail messages to an e-mail server are unauthenticated and can be easily forged, their content needs to be analysed and classified in addition to simple checks such as the "To" and "From" addresses. Modern anti-spam tools use heuristics and actually process the content of the message. Since different people receive different types of messages, the best of such systems adapt their heuristics to personalize them for each user.

Disabling Sendmail

FreeBSD is shipped with Sendmail as the default e-mail server system. Sendmail is enough for very simple usage, but quickly gets very complicated when additional features need to be configured. Since the goal of this tutorial is to install Postfix, Sendmail needs to be disabled before Postfix can function by adding the following line to `/etc/rc.conf`:

```
sendmail_enable="NONE"
```

A reboot is recommended to stop Sendmail listening on various network ports.

Installing and configuring Postfix

Postfix can be installed from a package with a command such as:

```
# pkg install postfix
```

When asked about activating Postfix in the `/etc/mail/mailer.conf`, answer "y".

Postfix configuration files are located in `/usr/local/etc/postfix`.

Its main configuration file is `main.cf`, and this is the file which will be modified in the next steps.

Postfix is very careful about which e-mail to receive, and the first line of defense is specifying the domain for which it will act as an SMTP end-point server. By default, this domain will be extracted from the server's host name, specified in the `myhostname` directive, such as:

```
myhostname = mail.example.com
```

The next step is to configure the domains which will be accepted as "local" for mail delivery:

```
mydestination = $myhostname, localhost.$mydomain,
   localhost, $mydomain
```

A common configuration is for the SMTP server to be configured to accept e-mail from clients in the same local network without authentication. This is convenient for the users as it skips the requirement for SMTP login, but can escalate into a problem if one of the local machines gets taken over by malware which will (ab) use the server for sending spam e-mail.

To configure Postfix to accept connections from IP addresses in the same subnets as the server as "trusted", configure `mynetworks_style`: mynetworks_style = subnet.

This method will automatically detect the server's IP addresses and subnets. In case manual configuration is required, use the `mynetworks` directive. Some setups (for example, servers behind ADSL connections) require that e-mail is not routed directly (which is the default behaviour) but is always relayed by the ISP's e-mail server. In this case (and only in this case), use the `relayhost` directive to specify the "upstream" e-mail server:

```
relayhost = mail.myisp.com
```

Such setups usually require that the connection to the upstream server is authenticated with the username and password provided by the ISP. This can be achieved by adding the following lines to the `main.cf` file:

```
smtp_sasl_auth_enable = yes
smtp_sasl_password_maps = hash:/usr/local/etc/postfix/
   sasl_passwd
smtp_sasl_security_options =
```

The password file specified in the `smtp_sasl_password_maps` needs to be created with the following content:

```
mail.myisp.com username:password
```

This file can contain multiple lines, each specifying a server's name and the username and password used when connecting to it. Usually, only one line is required. Since this is a security sensitive file, you should adjust its file access permissions as needed.

This text file needs to be converted into Postfix's "hashmap" format by issuing the following command:

```
# postmap hash:/usr/local/etc/postfix/sasl_passwd
```

For historical reasons, the /etc/aliases file needs to be parsed and a hashmap file created by issuing the `newaliases` command:

```
# newaliases
```

Lastly, Postfix needs to be enabled in `/etc/rc.conf` by adding the following line:

```
postfix_enable = "YES"
```

It can be immediately started by issuing a command such as:

```
# service postfix start
```

Sending an example e-mail message to test Postfix

The built-in FreeBSD program named "mail" can be used to send an example e-mail message. You can specify the e-mail Subject value with the "-s" argument, and when the program starts, it will read a message directly from the console. You should write an example message and end it with Ctrl-D:

```
> mail -s "A test message" ivoras@example.com
```

An example message here.

```
[Ctrl-D]
```

The Ctrl-D keyboard combination will create an "end-of-file" signal to the reading program. To check if the e-mail was successfully delivered, check the `/var/log/maillog` file and check that a file was created for the user given in the above command in the `/var/mail` directory.

Installing SpamAssassin

SpamAssassin is a framework for very configurable and adaptable e-mail analysis. It has multiple optional plugins which enhance its core functionality. SpamAssassin can be installed with the following command:

```
# pkg install spamassassin
```

Its configuration files are located in `/usr/local/etc/mail/spamassassin`.

After the installation, you should run the sa-update utility to refresh the database of common spam patterns. You can do this by adding a line like the following into `/etc/crontab`:

```
* * * * 1 root /usr/local/bin/sa-update
```

Enable the SpamAssassin daemon by adding the following line to `/etc/rc.conf`:

```
spamd_enable="YES"
```

Start the daemon with a command such as:

```
# service sa-spamd start
```

Integrating SpamAssassin with Postfix

A small helper shell script is required to integrate SpamAssassin with Postfix. For this for this article, we will name it `/root/spamfilter.sh` and give it the following content:

```
#!/bin/sh
SENDMAIL=/usr/local/sbin/sendmail
SPAMASSASSIN=/usr/local/bin/spamc
# logger <<<"Spam filter piping
to SpamAssassin, then to: $SENDMAIL $@"
${SPAMASSASSIN} | ${SENDMAIL} "$@"
exit $?
```

Don't forget to make the file executable.

Next, modify the `/usr/local/etc/postfix/master.cf` file. The first "smtp" line needs to be changed to:

```
smtp
inet n - - - - smtpd -o
content_filter=spamfilter
```

In addition to this, one new line needs to be added:

```
spamfilter
unix - n n -
- pipe
flags=Rq user=spamd
argv=/root/spamfilter.sh -oi -f ${sender} ${recipient}
```

After these modifications, Postfix needs to be restarted:

```
# service postfix restart
```

Testing SpamAssassin

The configuration described in master.cf enables spam filtering on the "smtp" service which is bound to the TCP port 25. This means that e-mail sent directly through Unix local delivery will not be filtered and SpamAssassin cannot be tested with the "mail" command. To test SpamAssassin, you need to send e-mail through the SMTP port 25.

There is a special string which can be used for testing. If this string is found in the message by SpamAssassin, the message will receive 1000 spam points and be marked as spam. This string is:

```
XJS*C4JDBQADN1.NSBN3*2IDNEN*GTUBE-STANDARD-ANTI-UBE-TEST-
    EMAIL*C.34X
```

Installing and configuring Dovecot as an IMAP server

IMAP is a protocol for e-mail message retrieval, used by user-facing applications to fetch messages and offer them in some sort of user interface. In contrast to the old POP3 protocol, IMAP offers a unified "view" of the message database on the server (messages are usually not deleted from the server when retrieved), supports subdirectories of the main mailbox and offers some rudimentary ability to share mailbox folders between different users.

Dovecot is one of the most popular IMAP server applications. It is well-written and extensible, and cooperates well with Postfix.

How e-mail is delivered

An SMTP server (operating in one of the roles called a Mail Transfer Agent or Mail Submission Agent) accepts a message and then either relays it to another server or attempts to deliver it into a "local" mailbox (meaning a mailbox on the server where the SMTP server is running). During the process of the delivery, the message may be processed in a number of ways, for example, by scanning it for spam (as seen in the previous tutorial). The delivery is performed by a module of the SMTP server called a "delivery agent." A "mailbox" is usually a single text file to which all the e-mail is concatenated, together with some special lines which delimit it. An alternative standard is called "maildir" which stores each message in its own file, in a special directory structure. Once the message is safely written to this storage, the SMTP server's job is done.

Another type of server, for example, the IMAP server, implements a protocol by which an application for reading e-mail retrieves and presents these messages to the user (such applications are called Mail User Agents). The IMAP server needs to find the messages and is configured completely separately from the SMTP server. In addition to simply presenting the messages from the mailbox file through the IMAP protocol, Dovecot can perform some interesting additional features, especially when integrated with Postfix. By default, Dovecot will maintain an indexed database of messages, which greatly speeds up all operations with them, but also contains a specialised delivery agent which is smarter than a plain SMTP delivery agent and can filter certain types of messages into certain IMAP folders.

Installing Dovecot

The current version of Dovecot is available in the package named `dovecot2`, and the Sieve message filtering module is in the package named `dovecot-pigeonhole`. Those packages should be installed with the usual `pkg` command:

```
# pkg install dovecot2 dovecot-pigeonhole
```

Its configuration files' directory is `/usr/local/etc/dovecot`, but the directory is empty after the installation and needs to be populated first. The directory:

```
/usr/local/share/doc/dovecot/example-config
```

contains example configuration files, all of which should be copied into Dovecot's configuration directory (preserving the directory structure, i.e. the `conf.d` subdirectory). In the same way, copy the files from `/usr/local/share/doc/dovecot-pigeonhole/example-config/conf.d` into Dovecot's `conf.d` directory.

For a basic configuration, you should modify the following files, and make sure that the specific configuration lines are present and uncommented in them:

```
dovecot.conf
```

Enable only IMAP with the following line:

```
protocols = imap
conf.d/10-auth.conf
```

Disable non-encrypted plaintext logins, include the default system authentication mechanism, and specify the correct plugins directory:

```
disable_plaintext_auth = yes
!include auth-system.conf.ext
mail_plugin_dir = /usr/local/lib/dovecot
conf.d/10-mail.conf
```

Specify that the default user mailbox is found in `/var/mail`, but that the additional mailbox files for IMAP folders will be in the directory `~/mail` for each user separately:

```
mail_location = mbox:~/mail:INBOX=/var/mail/%u
```

You should also modify the locking methods configuration to skip the “dotlock” method which is tricky to use securely when the default mailbox is located in `/var/mail`:

```
mbox_read_locks = fcntl
mbox_write_locks = fcntl
conf.d/10-ssl.conf
```

Specify where the TLS certificates are and which cipher suite to use (the same ones used for Apache):

```
ssl_cert =</var/ssl/ivoras.net.crt
ssl_key =</var/ssl/ivoras.net.key
ssl_cipher_list =
!ADH:!EXPORT:!SSLv2:EECDH+aRSA+AESGCM:EECDH+aRSA+RC4:RC4+R
   SA:+HIGH:+MEDIUM:+LOW
conf.d/15-lda.conf
```

Specify that the local delivery agent will use the Sieve plugin:

```
protocol lda {
mail_plugins = $mail_plugins sieve
}
conf.d/15-mailboxes.conf
```

In this file, modify all of the mailbox sections and add a line:

```
auto = subscribe
```

to each uncommented section.

```
conf.d/90-sieve.conf
```

Enable some convenient Sieve extensions: for modifying the e-mail headers and for manipulating IMAP message flags:

```
sieve_extensions = +editheader +imap4flags
```

Using the Dovecot local delivery agent in Postfix
In order to make use of the Sieve filtering plugin, Postfix needs to be configured to pass the e-mail which would be delivered locally to Dovecot’s local delivery agent module. This is done very simply, with the following line in Postfix’s `main.cf` file:

```
mailbox_command = /usr/local/libexec/dovecot/dovecot-lda
   -f “$SENDER” -a
“$RECIPIENT”
```

Restarting Postfix and Dovecot
Before using Dovecot, enable it in `/etc/rc.conf` with a line such as the following:

```
dovecot_enable=”YES”
```

Postfix and Dovecot can be restarted by issuing the following commands:

```
# service postfix restart
# service dovecot restart
```

Creating Sieve rules
Sieve rules can be created globally, for all users, or with user-specific scripts. This tutorial will describe the per-user scenario (for the global case you should look at the `sieve_default` directive in `90-sieve.conf`). By default, Dovecot-Pigeonhole will try to find a Sieve script file named `~/.dovecot.sieve` in each user’s home directory.

Sieve has a programming language which is designed for simple rule-based e-mail processing. It is a powerful language which can perform many actions, but the most common uses of Sieve are for filtering spam messages and for sorting e-mail messages into separate IMAP folders.

An example Sieve script can be as follows:

```
require [“fileinto”, “envelope”, “imap4flags”, “regex”,
   “editheader”,”variables”];
if header :contains “X-Spam-Flag””YES” {
fileinto “Junk”;
} elsif address :contains “to””ivoras@example.com” {
addheader “Importance””High”;
addflag “$label3”;
} elsif address :is “to””mailing-list@example.com” {
fileinto “mailing-list”;
} else {
keep;
}
```

The above script will perform the following actions on each message:

1. If the e-mail headers contain the SpamAssassin’s flag, save the message into the “Junk” folder
2. If the message’s “To” header mentions my address directly, mark the message as important (this is so that messages which contain my address only in the

CC field or which are passed through mailing lists are marked as “less important”)
3. If the message is sent to a specific mailing list, save it to a separate folder

Sieve can do much more, and you should study the examples given at goo.gl/QGBgDS.

You can test that everything is working by sending some e-mail which will match the above Sieve rules (after restarting Postfix and Dovecot). It is easy to make syntax errors in Sieve, but luckily the Pigeonhole Sieve module will log such errors into a file named `~/.dovecot.sieve.log`.

Be sure to check `/var/log/maillog` for error messages!

Connecting to the IMAP server

When configuring an e-mail reading application, you should connect to the IMAP server on the standard port 143, and use TLS for secure network traffic, which includes logins.

Installing and configuring the RoundCube webmail application

RoundCube is a “normal” PHP application which offers the user the ability to login with a username and password to an IMAP server. It collects the messages and folder from the IMAP server and displays them in a nice graphical interface. It also uses a database for storing miscellaneous information such as user preferences and the contacts list (address book), but the amount of information stored to the database is very small (it does NOT store e-mail messages in the database; the messages are only stored on the IMAP server).

Installing and configuring RoundCube

RoundCube requires the following PHP module to be installed in addition to those installed for ownCloud:

php5-filter

Similarly to how ownCloud was installed in Tutorial #6, this tutorial will explain how to install RoundCube from current official sources, put into the `/srv/www/roundcube` directory.

As a first step, download the source archive from *http://roundcube.net/download/* and put it into `/srv/www`:

```
# cd /srv/www
# fetch
http://sourceforge.net/projects/roundcubemail/files/
   roundcubemail/1.0.3/roundcubemail-1.0.3.tar.gz/download
# tar xzf download
# mv roundcubemail-1.0.3 roundcube
# chown -R ivoras roundcube
```

The application needs to write logs and temporary files, so the appropriate paths should be allowed to be written by the web server:

```
# chgrp -R www roundcube/temp roundcube/logs
# chmod -R g+rw roundcube/temp roundcube/logs
```

RoundCube needs to be configured by copying the `config.inc.php.sample` into `config.inc.php` in the config subdirectory, and modifying the following configuration variables:

```
$config['db_dsnw'] = 'mysql://roundcube@localhost/
   roundcube';
$config['des_key'] = '1mka9f84mrandomrandom123';
$config['mime_types'] = '/usr/local/etc/apache24/mime.types';
$config['default_host'] = 'tls://localhost';
$config['preview_pane'] = true;
$config['preview_pane_mark_read'] = 2;
$config['enable_installer'] = true;
```

The first line configures the database configuration. For it to be valid, you should create the roundcube database in MySQL and grant the roundcube user all rights on it:

```
# mysql
```

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 1

Server version: 5.5.40 Source distribution

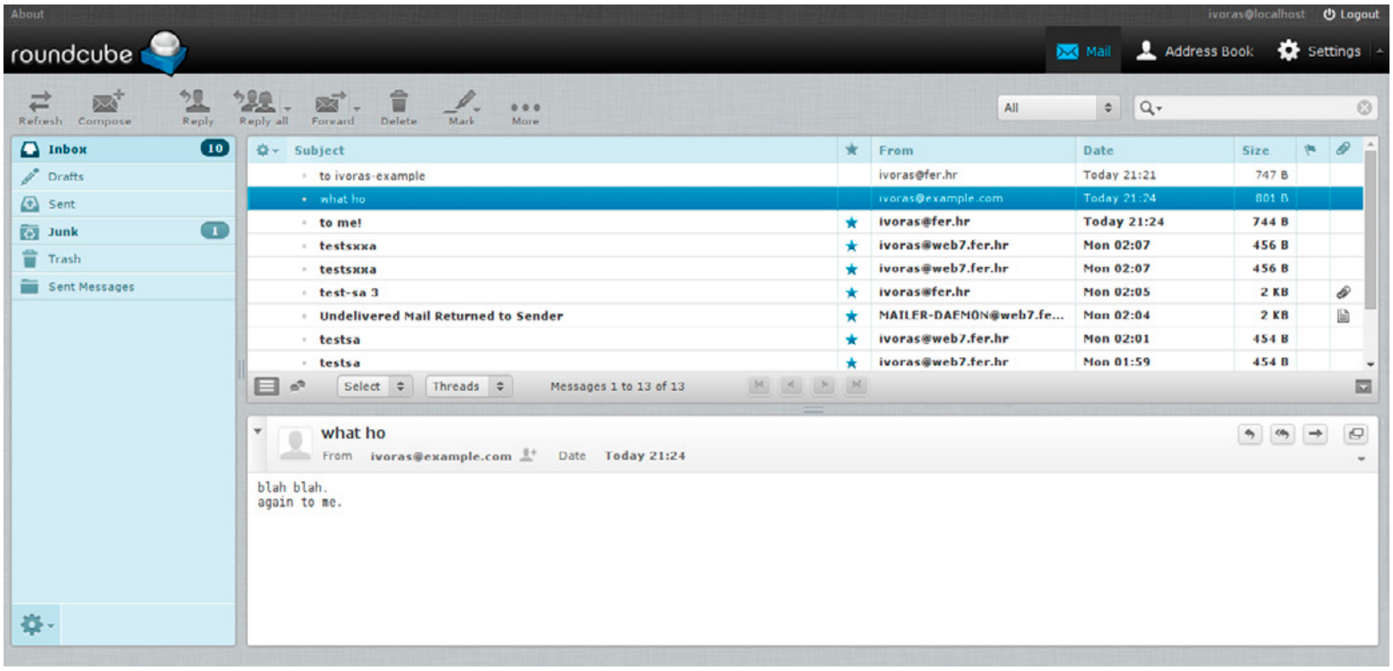
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Type ‘help;’ or ‘\h’ for help. Type ‘\c’ to clear the current input statement.

```
mysql> create database roundcube;
Query OK, 1 row affected (0.00 sec)
mysql> grant all on roundcube.* to
'roundcube'@'localhost';
Query OK, 0 rows affected (0.02 sec)
```

The second line in the configuration file needs to specify a unique key used by RoundCube to securely transmit some session-related information. The third line enables the built-in installation process, and will need to be removed before RoundCube is used in production.



Subject	★	From	Date	Size
to ivoras-example		ivoras@fer.hr	Today 21:21	747 B
what ho		ivoras@example.com	Today 21:24	801 B
to me!	★	ivoras@fer.hr	Today 21:24	744 B
testsxxa	★	ivoras@web7.fer.hr	Mon 02:07	456 B
testsxxa	★	ivoras@web7.fer.hr	Mon 02:07	456 B
test-sa 3	★	ivoras@fer.hr	Mon 02:05	2 KB
Undelivered Mail Returned to Sender	★	MAILER-DAEMON@web7.fe...	Mon 02:04	2 KB
testsa	★	ivoras@web7.fer.hr	Mon 02:01	454 B
testsa	★	ivoras@web7.fer.hr	Mon 01:59	454 B

what ho
From ivoras@example.com Date Today 21:24

```
blah blah.
again to me.
```

Figure 1.

Next, Apache's virtual host configuration for the TSL host on port 443 needs to be modified to allow access to the newly installed PHP application:

```
Alias /mail "/srv/www/roundcube"
<Directory "/srv/www/roundcube">
Options ExecCGI FollowSymLinks
AddHandler fcgid-script php
FCGIWrapper /usr/local/bin/php-cgi
.php
DirectoryIndex index.php
AllowOverride All
Require all granted
</Directory>
```

The reason why it is necessary to allow access to RoundCube only from a SSL-enabled virtual host is because, like ownCloud, it requires a login through the web page, so its username and password need to be protected.

RoundCube also requires some configuration changes to PHP. By default, there is no `php.ini` in a freshly installed PHP on FreeBSD, but there are two example files named `php.ini-development` and `php.ini-production` in `/usr/local/etc`. You should copy the production version of the file into the `php.ini` file and change the following lines:

```
date.timezone = Europe/Zagreb
```

Next, RoundCube's internal installation process needs to be started by visiting the `/installer` path in its installation, such as `https://example.com/mail/installer`. If everything is OK, you should arrive at a button to initialize the database, which you should click.

Conclusions

After database tables are created, you can optionally run the tests for the SMTP and IMAP servers (the default configuration assumes that they are both on localhost), then remove the installer line from config.php and the installer directory from RoundCube sources, then visit its main URL at *https://example.com/mail*.

ABOUT THE AUTHOR

Ivan Voras is a FreeBSD developer and a long-time user, starting with FreeBSD 4.3 and throughout all the versions since. In real life he is a researcher, system administrator and a developer, as opportunity presents itself, with a wide range of experience from hardware hacking to cloud computing. He is currently employed at the University of Zagreb Faculty of Electrical Engineering and Eomputing and lives in Zagreb, Croatia. You can follow him on his blog in English at http://ivoras.net/blog or in Croatian at http://hrblog.ivoras.net/, as well as Google+ at https://plus.google.com/+IvanVoras.

Big Data Gets Real in Boston!

People are talking about BigData TechCon!

"Big Data TechCon is a great learning experience and very intensive."
—Huaxia Rui, Assistant Professor, University of Rochester

"Get some sleep beforehand, and divide and conquer the packed schedule with colleagues."
—Paul Reed, Technology Strategy & Innovation, FIS

"Worthwhile, technical, and a breath of fresh air."
—Julian Gottesman, CIO, DRA Imaging

"Big Data TechCon is definitely worth the investment."
—Sunil Epari, Solutions Architect, Epari Inc.

BigData TECHCON

April 26-28, 2015

Seaport World Trade Center Hotel

Big Data TechCon is the HOW-TO technical conference for professionals implementing Big Data solutions at their company

Come to Big Data TechCon to learn the best ways to:

- Process and analyze the real-time data pouring into your organization
- Learn how to extract better data analytics and predictive analysis to produce the kind of actionable information and reports your organization needs.
- Come up to speed on the latest Big Data technologies like Yarn, Hadoop, Apache Spark and Cascading
- Understand HOW to leverage Big Data to help your organization today

www.BigDataTechCon.com

A BZ Media Event

The Basics of The GDB Debugger

CARLOS NEIRA

To be able to inspect a program more easily, we need to have the symbol table available for the program we intend to debug; this is accomplished by using the –g flag of the compiler we are going to use (we could also debug it without the –g flag but it is really cumbersome sometimes). In our case we will use FreeBSD 10 as the platform and the clang compiler that comes with it.

After a program is compiled using the –g flag we are able to peek inside it using the gdb debugger. To start a debugging session. All you need to type is the following:

```
# gdb <program_name>
```

And we will see a (gdb) prompt. That means that we are ready to start typing gdb commands (Figure 1).

Or if the program we need to debug is currently running, we must type:

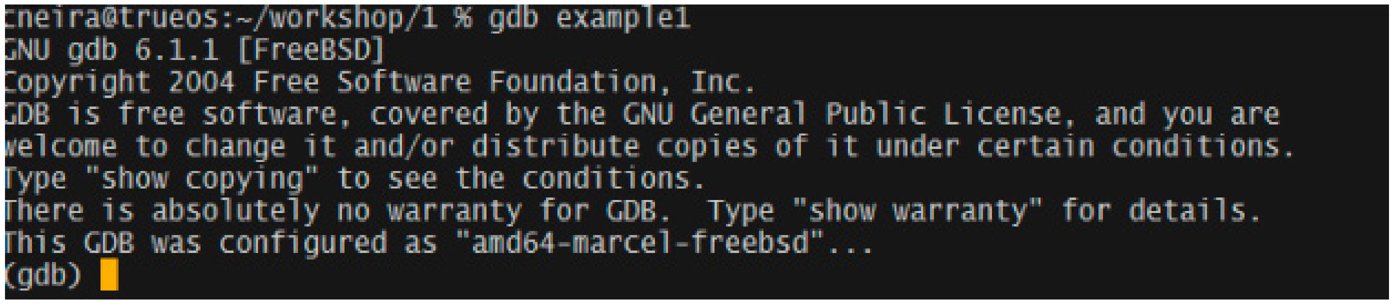
```
#gdb
#(gdb) attach <pid of running program>
```

Let's start with some basic commands and inspect a running application. Dor this example I have selected this application *http://freeciv.wikia.com/wiki/Main_Page*.

"Freeciv is a Free and Open Source empire-building strategy game inspired by the history of human civilization. The game commences in prehistory and your mission is to lead your tribe from the Stone Age to the Space Age…"

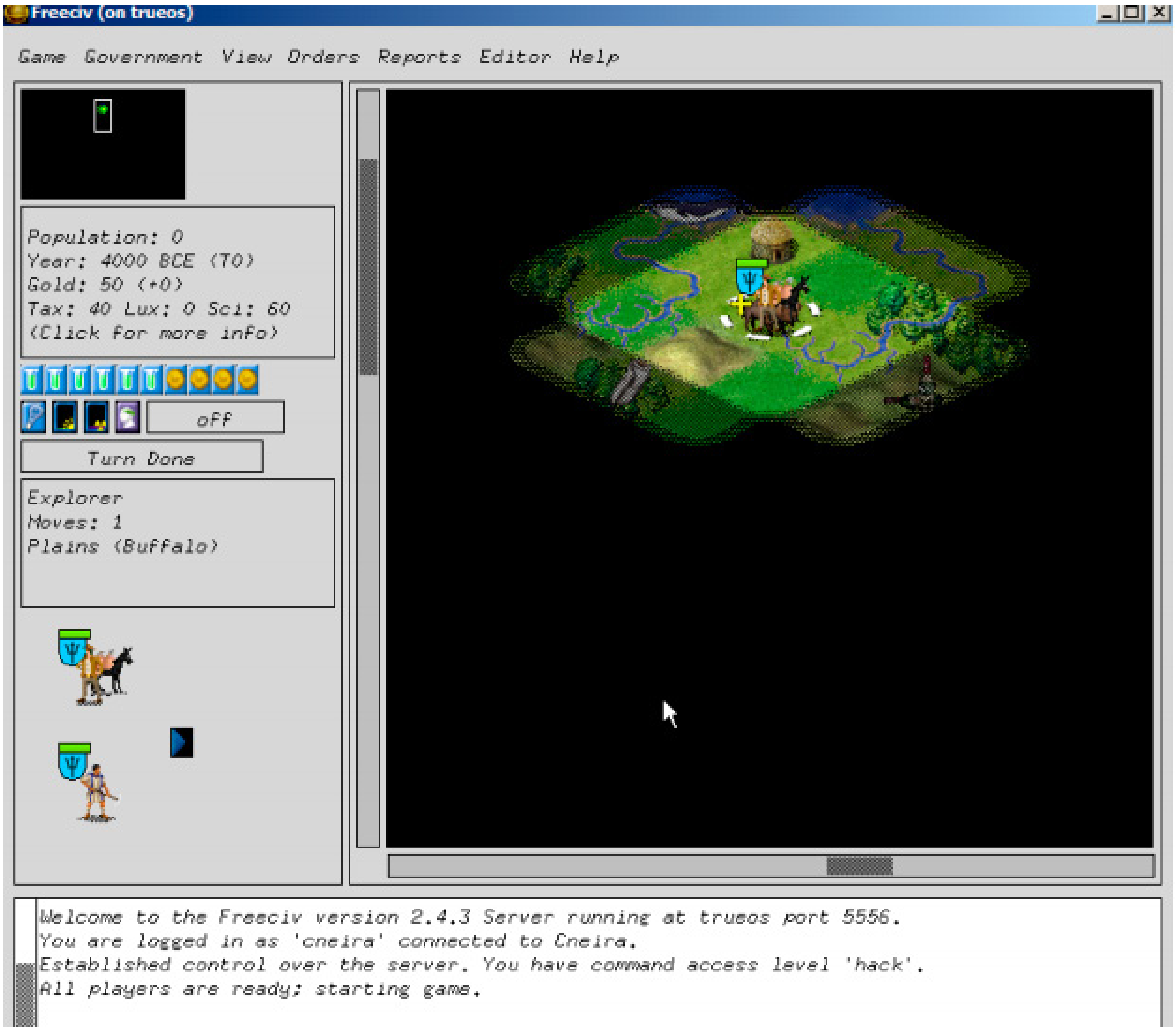
We will inspect the game structures at runtime with gdb. Let's follow these steps:

- Edit `/etc/make.conf` and add the line `WITH _ DEBUG=yes` (this will not strip your binaries so you will have the symbol table and also add the debug flags to the compiler when compiling the sources of your ports)



```
cneira@trueos:~/workshop/1 % gdb example1
GNU gdb 6.1.1 [FreeBSD]
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for details.
This GDB was configured as "amd64-marcel-freebsd"...
(gdb)
```

Figure 1. *GDB example*

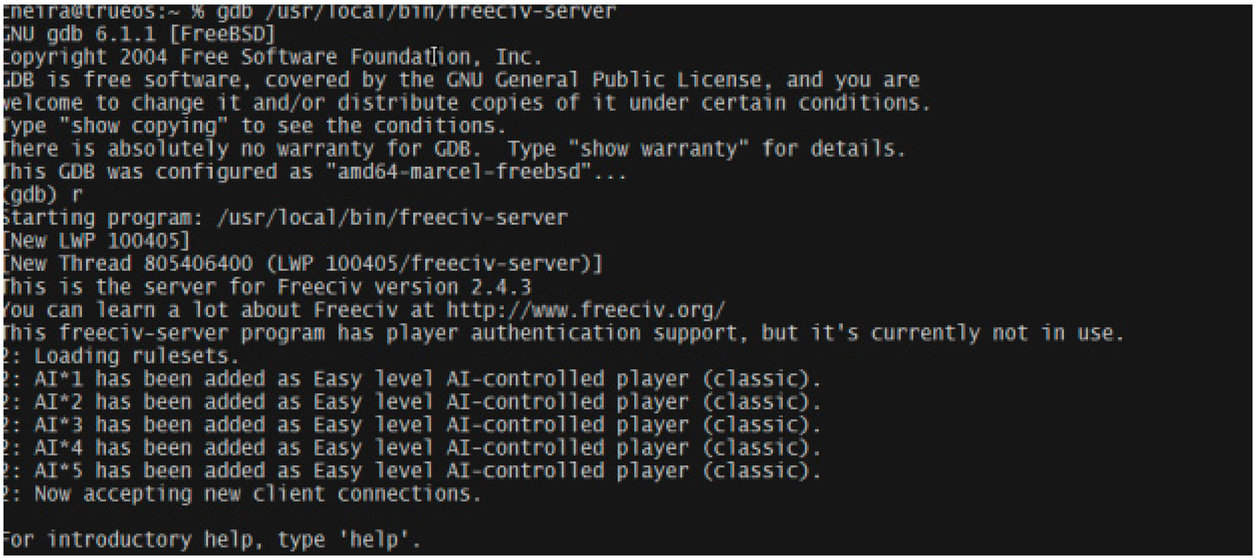


```
Population: 0
Year: 4000 BCE (T0)
Gold: 50 (+0)
Tax: 40 Lux: 0 Sci: 60
(Click for more info)
```

```
Explorer
Moves: 1
Plains (Buffalo)
```

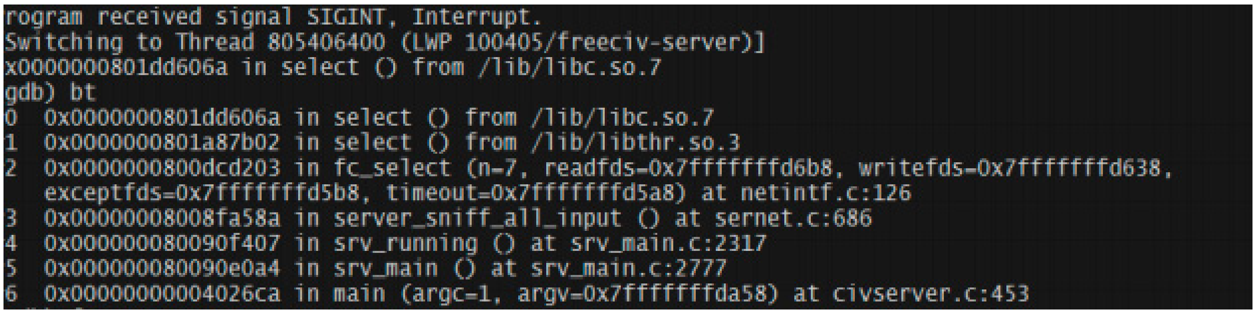
```
Welcome to the Freeciv version 2.4.3 Server running at trueos port 5556.
You are logged in as 'cneira' connected to Cneira.
Established control over the server. You have command access level 'hack'.
All players are ready; starting game.
```

Figure 2. *Joined the local game*



```
cneira@trueos:~ % gdb /usr/local/bin/freeciv-server
GNU gdb 6.1.1 [FreeBSD]
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for details.
This GDB was configured as "amd64-marcel-freebsd"...
(gdb) r
Starting program: /usr/local/bin/freeciv-server
[New LWP 100405]
[New Thread 805406400 (LWP 100405/freeciv-server)]
This is the server for Freeciv version 2.4.3
You can learn a lot about Freeciv at http://www.freeciv.org/
This freeciv-server program has player authentication support, but it's currently not in use.
2: Loading rulesets.
2: AI*1 has been added as Easy level AI-controlled player (classic).
2: AI*2 has been added as Easy level AI-controlled player (classic).
2: AI*3 has been added as Easy level AI-controlled player (classic).
2: AI*4 has been added as Easy level AI-controlled player (classic).
2: AI*5 has been added as Easy level AI-controlled player (classic).
2: Now accepting new client connections.

For introductory help, type 'help'.
```

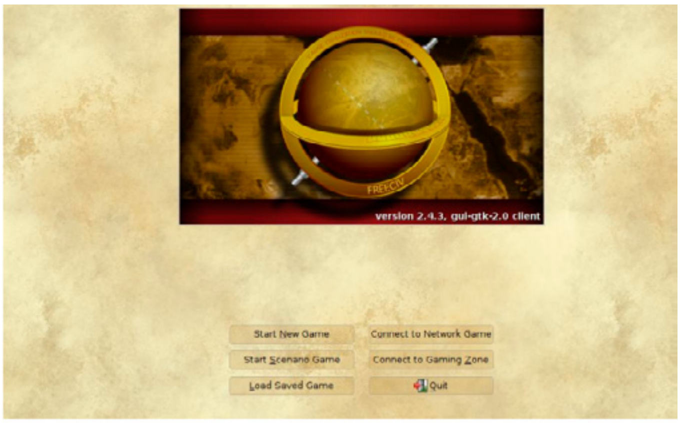
Figure 3. *The GNU General Public License*



```
rogram received signal SIGINT, Interrupt.
Switching to Thread 805406400 (LWP 100405/freeciv-server)]
x0000000801dd606a in select () from /lib/libc.so.7
gdb) bt
0  0x0000000801dd606a in select () from /lib/libc.so.7
1  0x0000000801a87b02 in select () from /lib/libthr.so.3
2  0x0000000800dcd203 in fc_select (n=7, readfds=0x7fffffffd6b8, writefds=0x7fffffffd638,
   exceptfds=0x7fffffffd5b8, timeout=0x7fffffffd5a8) at netintf.c:126
3  0x00000008008fa58a in server_sniff_all_input () at sernet.c:686
4  0x000000080090f407 in srv_running () at srv_main.c:2317
5  0x000000080090e0a4 in srv_main () at srv_main.c:2777
6  0x00000000004026ca in main (argc=1, argv=0x7fffffffda58) at civserver.c:453
```

Figure 4. *To continue the execution*

- Install freeciv from ports
- Start the freeciv server and client (freeciv-server and freeciv-gtk2)
- Join your local game (Figure 2)



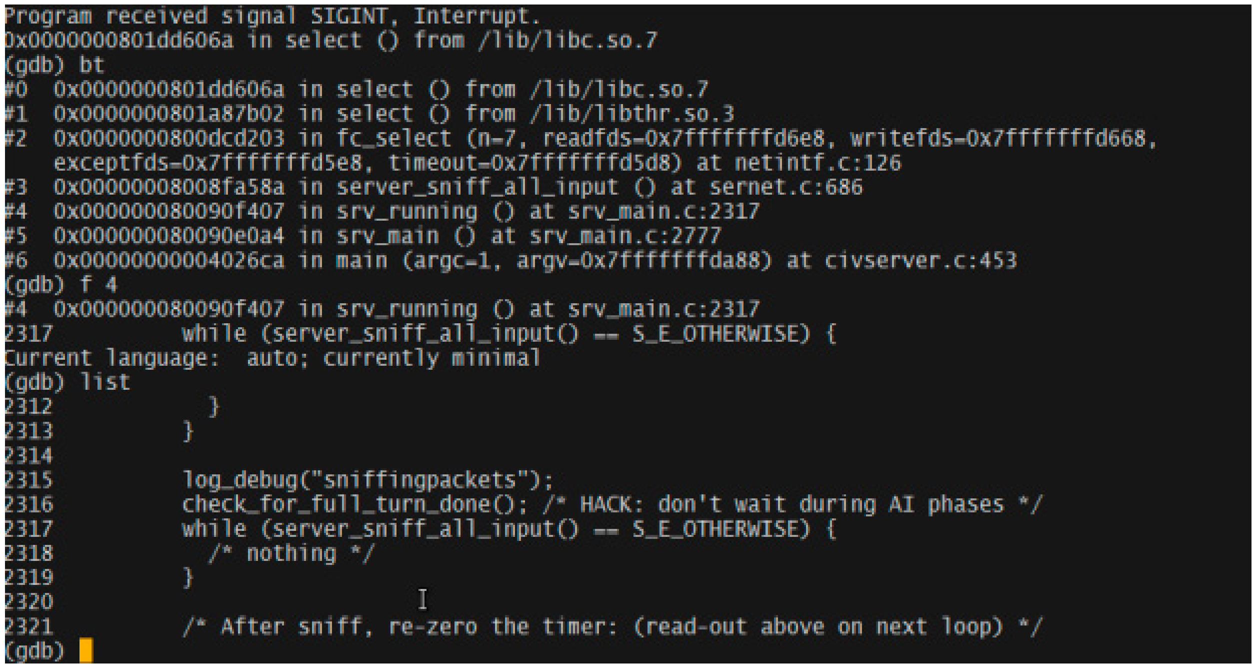
Now we will use our first gdb command:

```
# gdb /usr/local/bin/freeciv-server
```

As we don't know anything about how Freeciv works, we will press CTRL-C. This will interrupt the program and we will take it from there. For starters, let's interrupt and see where we are. If we want to continue the execution, we type 'continue' or 'c' (Figure 4).

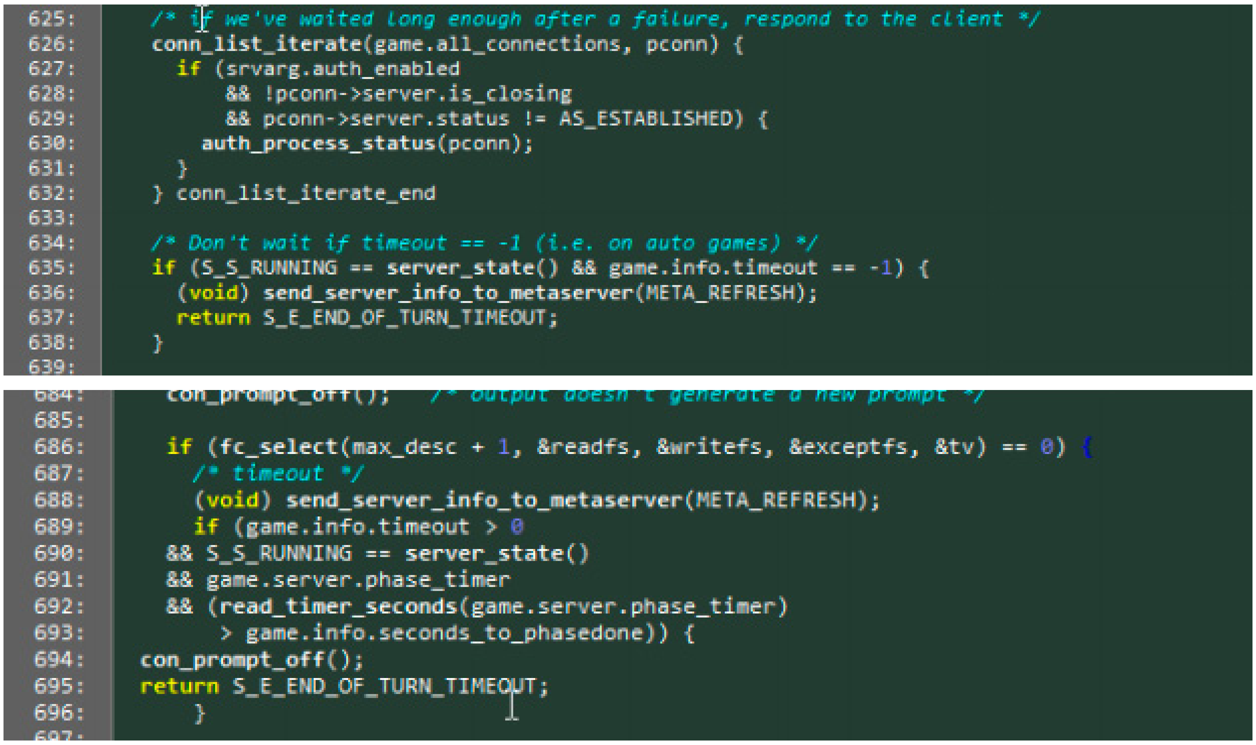
Figure 5 is a screenshot from the client program freeciv-gtk2; we need to join our local game as we are going to debug the server (Figure 5).

The `#<num>` you see are the stackframes of simply called frames. When your program is started, the stack has only one frame, that of the function main. This is called the initial frame or the outermost frame. Each time a function is called, a new frame is made. Each time a function returns, the frame



```
Program received signal SIGINT, Interrupt.
0x0000000801dd606a in select () from /lib/libc.so.7
(gdb) bt
#0  0x0000000801dd606a in select () from /lib/libc.so.7
#1  0x0000000801a87b02 in select () from /lib/libthr.so.3
#2  0x0000000800dcd203 in fc_select (n=7, readfds=0x7fffffffd6e8, writefds=0x7fffffffd668,
    exceptfds=0x7fffffffd5e8, timeout=0x7fffffffd5d8) at netintf.c:126
#3  0x00000008008fa58a in server_sniff_all_input () at sernet.c:686
#4  0x000000080090f407 in srv_running () at srv_main.c:2317
#5  0x000000080090e0a4 in srv_main () at srv_main.c:2777
#6  0x00000000004026ca in main (argc=1, argv=0x7fffffffda88) at civserver.c:453
(gdb) f 4
#4  0x000000080090f407 in srv_running () at srv_main.c:2317
2317          while (server_sniff_all_input() == S_E_OTHERWISE) {
Current language:  auto; currently minimal
(gdb) list
2312            }
2313          }
2314
2315          log_debug("sniffingpackets");
2316          check_for_full_turn_done(); /* HACK: don't wait during AI phases */
2317          while (server_sniff_all_input() == S_E_OTHERWISE) {
2318            /* nothing */
2319          }
2320
2321          /* After sniff, re-zero the timer: (read-out above on next loop) */
(gdb) 
```

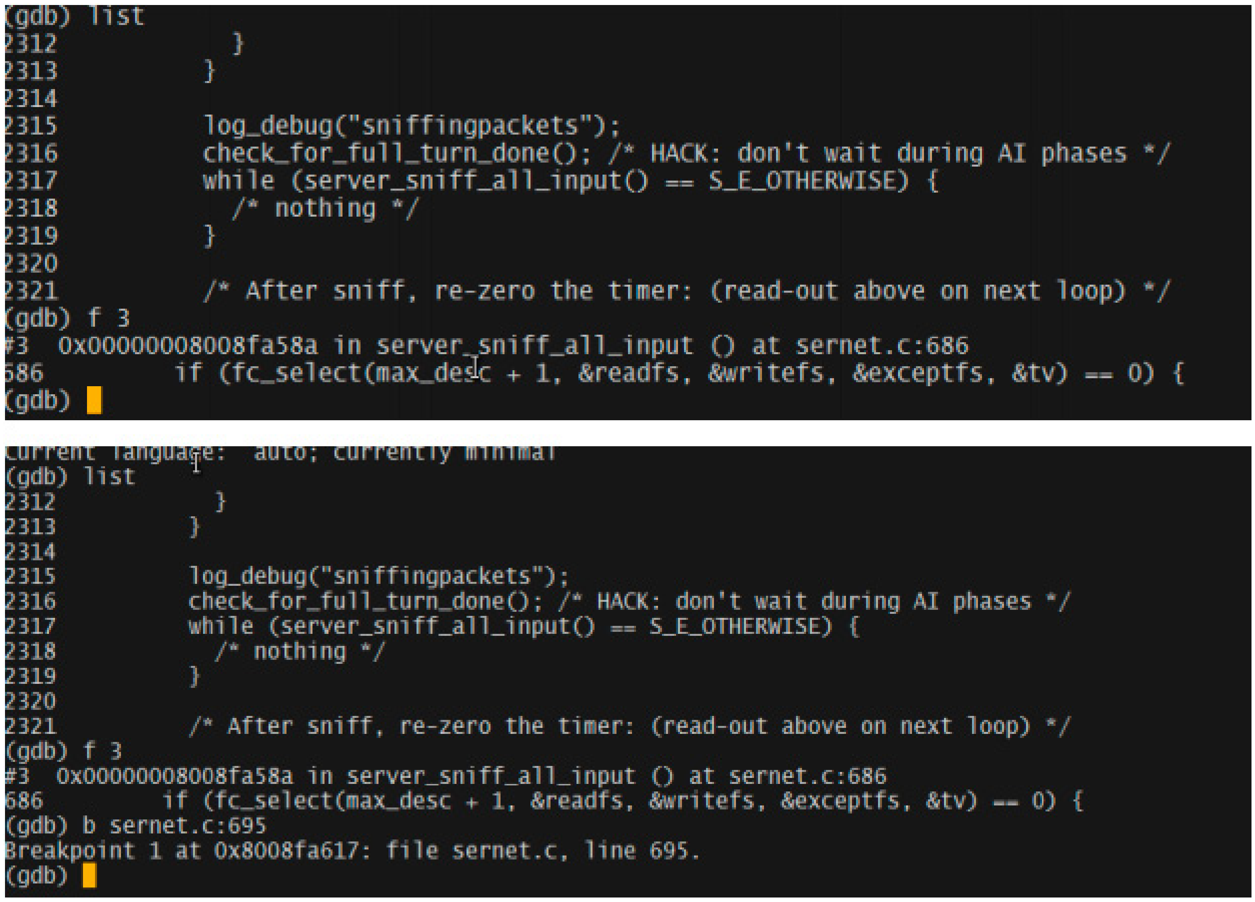
Figure 5. *The client program freecivgtk2*



```
625:      /* if we've waited long enough after a failure, respond to the client */
626:      conn_list_iterate(game.all_connections, pconn) {
627:        if (srvarg.auth_enabled
628:            && !pconn->server.is_closing
629:            && pconn->server.status != AS_ESTABLISHED) {
630:          auth_process_status(pconn);
631:        }
632:      } conn_list_iterate_end
633:
634:      /* Don't wait if timeout == -1 (i.e. on auto games) */
635:      if (S_S_RUNNING == server_state() && game.info.timeout == -1) {
636:        (void) send_server_info_to_metaserver(META_REFRESH);
637:        return S_E_END_OF_TURN_TIMEOUT;
638:      }
639:
```

```
684:      con_prompt_off();   /* output doesn't generate a new prompt */
685:
686:      if (fc_select(max_desc + 1, &readfs, &writefs, &exceptfs, &tv) == 0) {
687:        /* timeout */
688:        (void) send_server_info_to_metaserver(META_REFRESH);
689:        if (game.info.timeout > 0
690:      && S_S_RUNNING == server_state()
691:      && game.server.phase_timer
692:      && (read_timer_seconds(game.server.phase_timer)
693:          > game.info.seconds_to_phasedone)) {
694:    con_prompt_off();
695:    return S_E_END_OF_TURN_TIMEOUT;
696:        }
697:
```

Figure 6. *The innermost frame*



```
(gdb) list
2312                    }
2313                }
2314
2315                log_debug("sniffingpackets");
2316                check_for_full_turn_done(); /* HACK: don't wait during AI phases */
2317                while (server_sniff_all_input() == S_E_OTHERWISE) {
2318                  /* nothing */
2319                }
2320
2321                /* After sniff, re-zero the timer: (read-out above on next loop) */
(gdb) f 3
#3  0x000000008008fa58a in server_sniff_all_input () at sernet.c:686
686           if (fc_select(max_desc + 1, &readfs, &writefs, &exceptfs, &tv) == 0) {
(gdb) 
```

```
current language:  auto; currently minimal
(gdb) list
2312                    }
2313                }
2314
2315                log_debug("sniffingpackets");
2316                check_for_full_turn_done(); /* HACK: don't wait during AI phases */
2317                while (server_sniff_all_input() == S_E_OTHERWISE) {
2318                  /* nothing */
2319                }
2320
2321                /* After sniff, re-zero the timer: (read-out above on next loop) */
(gdb) f 3
#3  0x000000008008fa58a in server_sniff_all_input () at sernet.c:686
686           if (fc_select(max_desc + 1, &readfs, &writefs, &exceptfs, &tv) == 0) {
(gdb) b sernet.c:695
Breakpoint 1 at 0x8008fa617: file sernet.c, line 695.
(gdb) 
```

Figure 7. *The innermost frame*

for that function invocation is eliminated. If a function is recursive, there can be many frames for the same function. The frame for the function in which execution is actually occurring is called the innermost frame. This is the most recently created of all the stack frames that still exist. Let's go into frame 3; to do this we type either 'frame 3' or 'f 3' (Figure 6).

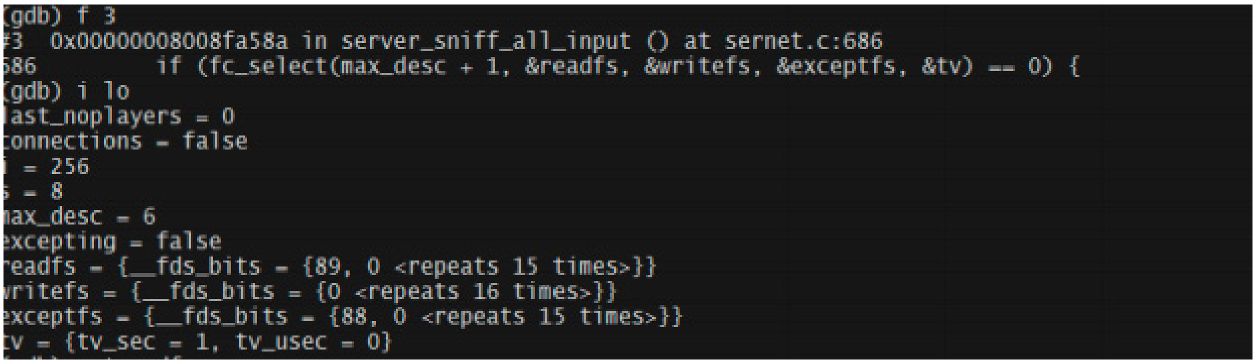
It seems that the server is going to send us end of turn. Let's make sure to set a break point, the format

```
is >
<break|b> <source.c>:<line number>
(gdb) b sernet.c:695
```

It seems we are wrong, let's interrupt again and inspect the data at this point (Figure 8).

Typing 'i lo' means info locals which will display all local variables in this frame and their values, which is pretty handy. Let's take a look at something easier to see. Sometimes in freeciv, another civilization will try to negotiate terms with us. Looking at the source code, we find the add_clause function in the diptreaty.c source code. That function will add a term which will make the other part accept or reject our terms (Figure 9 and Figure 10).

After playing a few minutes we hit this break point. At this point, we don't even know which civilization has

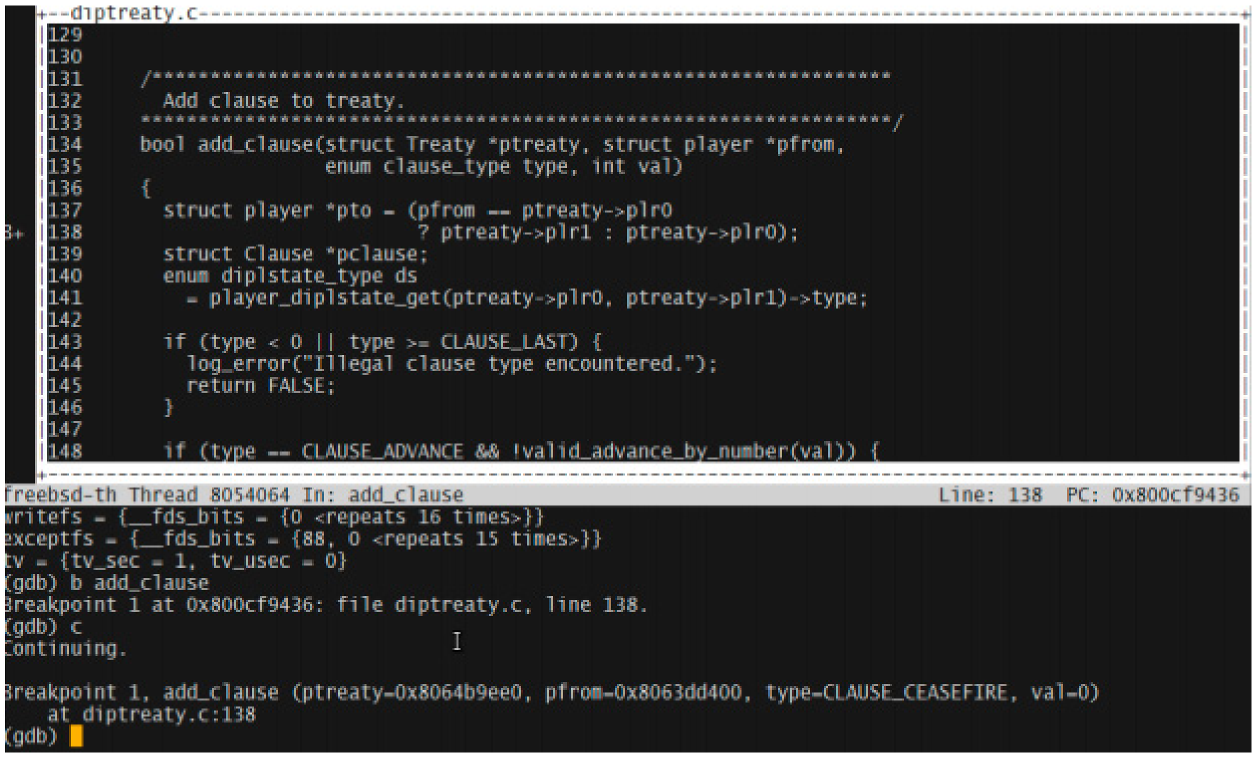


```
(gdb) f 3
#3  0x00000008008fa58a in server_sniff_all_input () at sernet.c:686
686         if (fc_select(max_desc + 1, &readfs, &writefs, &exceptfs, &tv) == 0) {
(gdb) i lo
ast_noplayers = 0
onnections = false
 = 256
 = 8
ax_desc = 6
xcepting = false
eadfs = {__fds_bits = {89, 0 <repeats 15 times>}}
ritefs = {__fds_bits = {0 <repeats 16 times>}}
xceptfs = {__fds_bits = {88, 0 <repeats 15 times>}}
v = {tv_sec = 1, tv_usec = 0}
```

Figure 8. *To interrupt*

```
/usr/home/cneira/workshop/freeciv-2.4.3/common
```

Figure 9. *The add_clause function*

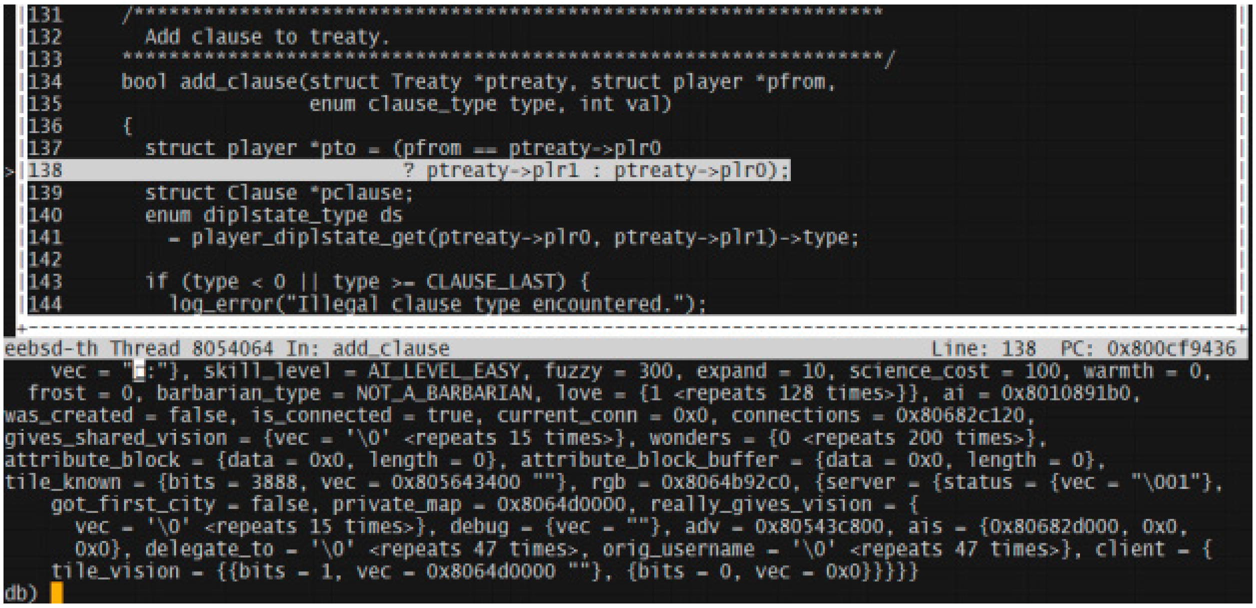


```
+--diptreaty.c---------------------------------------------------------------+


    /**************************************************************************
      Add clause to treaty.
    **************************************************************************/
    bool add_clause(struct Treaty *ptreaty, struct player *pfrom,
                    enum clause_type type, int val)
    {
      struct player *pto = (pfrom == ptreaty->plr0
B+                          ? ptreaty->plr1 : ptreaty->plr0);
      struct Clause *pclause;
      enum diplstate_type ds
        = player_diplstate_get(ptreaty->plr0, ptreaty->plr1)->type;

      if (type < 0 || type >= CLAUSE_LAST) {
        log_error("Illegal clause type encountered.");
        return FALSE;
      }

      if (type == CLAUSE_ADVANCE && !valid_advance_by_number(val)) {
+----------------------------------------------------------------------------+
freebsd-th Thread 8054064 In: add_clause                    Line: 138  PC: 0x800cf9436
ritefs = {__fds_bits = {0 <repeats 16 times>}}
xceptfs = {__fds_bits = {88, 0 <repeats 15 times>}}
v = {tv_sec = 1, tv_usec = 0}
(gdb) b add_clause
reakpoint 1 at 0x800cf9436: file diptreaty.c, line 138.
(gdb) c
ontinuing.

reakpoint 1, add_clause (ptreaty=0x8064b9ee0, pfrom=0x8063dd400, type=CLAUSE_CEASEFIRE, val=0)
    at diptreaty.c:138
(gdb) 
```

Figure 10. *The add_clause function*

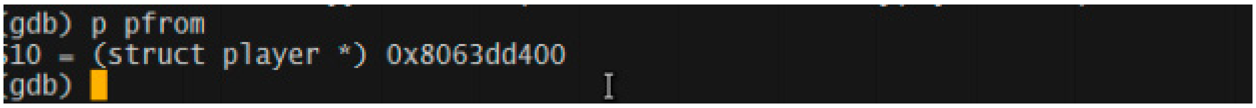
approached us to negotiate terms. Now we can know ahead of time as we set the breakpoint where the negotiation starts (Figure 11).

I assume the the negotiation civilization should be in the pfrom pointer (Figure12).



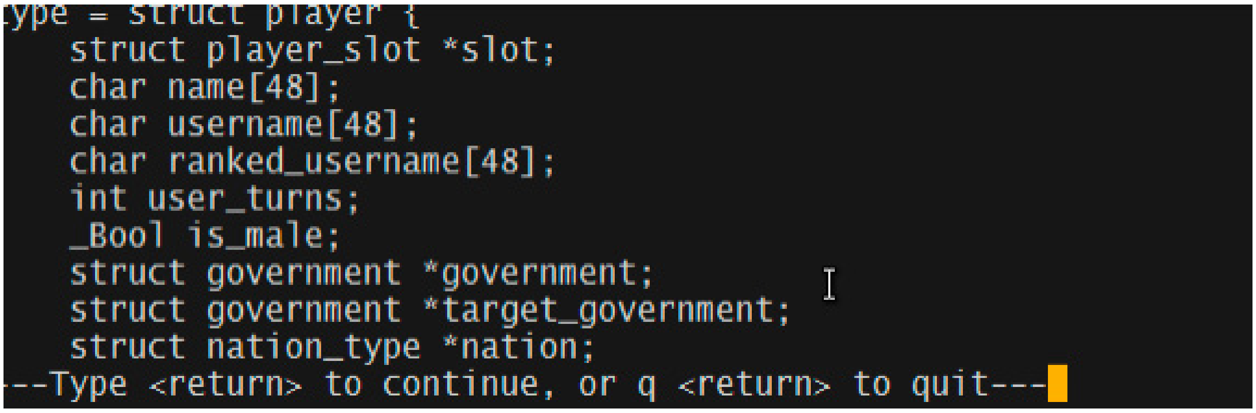
```
     /*******************************************************************
       Add clause to treaty.
     *******************************************************************/
     bool add_clause(struct Treaty *ptreaty, struct player *pfrom,
                     enum clause_type type, int val)
     {
       struct player *pto = (pfrom == ptreaty->plr0
                             ? ptreaty->plr1 : ptreaty->plr0);
       struct Clause *pclause;
       enum diplstate_type ds
         = player_diplstate_get(ptreaty->plr0, ptreaty->plr1)->type;

       if (type < 0 || type >= CLAUSE_LAST) {
         log_error("Illegal clause type encountered.");
eebsd-th Thread 8054064 In: add_clause                          Line: 138   PC: 0x800cf9436
    vec = "▯:"}, skill_level = AI_LEVEL_EASY, fuzzy = 300, expand = 10, science_cost = 100, warmth = 0,
  frost = 0, barbarian_type = NOT_A_BARBARIAN, love = {1 <repeats 128 times>}}, ai = 0x8010891b0,
was_created = false, is_connected = true, current_conn = 0x0, connections = 0x80682c120,
gives_shared_vision = {vec = '\0' <repeats 15 times>}, wonders = {0 <repeats 200 times>},
attribute_block = {data = 0x0, length = 0}, attribute_block_buffer = {data = 0x0, length = 0},
tile_known = {bits = 3888, vec = 0x805643400 ""}, rgb = 0x8064b92c0, {server = {status = {vec = "\001"},
    got_first_city = false, private_map = 0x8064d0000, really_gives_vision = {
      vec = '\0' <repeats 15 times>}, debug = {vec = ""}, adv = 0x80543c800, ais = {0x80682d000, 0x0,
      0x0}, delegate_to = '\0' <repeats 47 times>, orig_username = '\0' <repeats 47 times>}, client = {
    tile_vision = {{bits = 1, vec = 0x8064d0000 ""}, {bits = 0, vec = 0x0}}}}}
db)
```

Figure 11. *In the pfrom pointer*



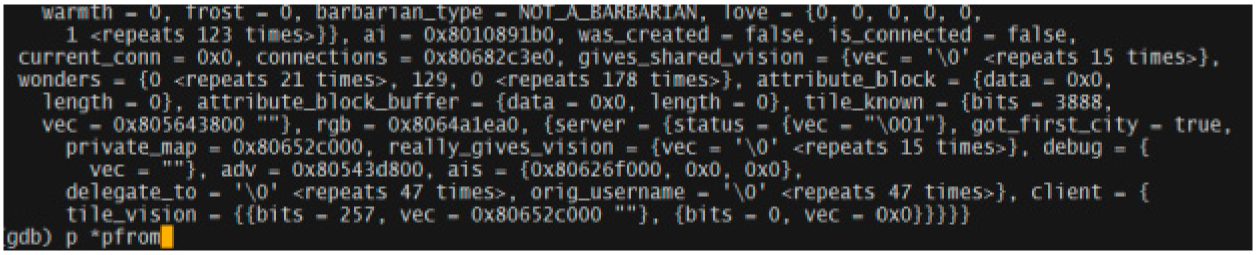
```
(gdb) p pfrom
$10 = (struct player *) 0x8063dd400
(gdb)
```

Figure 12. *In the pfrom pointer*



```
type = struct player {
    struct player_slot *slot;
    char name[48];
    char username[48];
    char ranked_username[48];
    int user_turns;
    _Bool is_male;
    struct government *government;
    struct government *target_government;
    struct nation_type *nation;
---Type <return> to continue, or q <return> to quit---
```

Figure 13. *In the pfrom pointer*



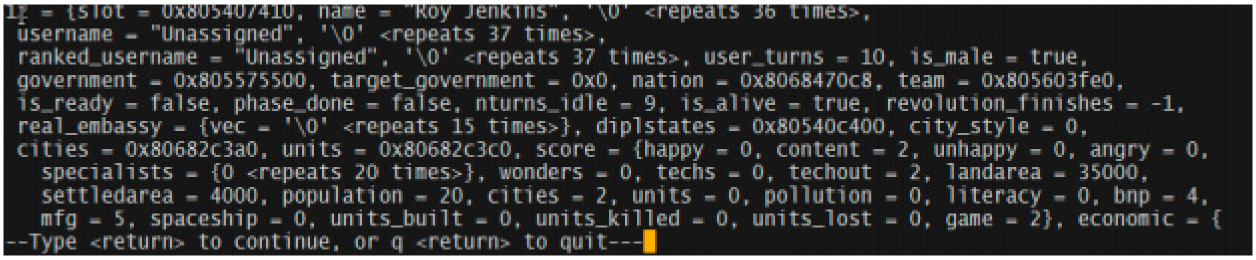
```
   warmth = 0, frost = 0, barbarian_type = NOT_A_BARBARIAN, love = {0, 0, 0, 0, 0,
     1 <repeats 123 times>}}, ai = 0x8010891b0, was_created = false, is_connected = false,
 current_conn = 0x0, connections = 0x80682c3e0, gives_shared_vision = {vec = '\0' <repeats 15 times>},
 wonders = {0 <repeats 21 times>, 129, 0 <repeats 178 times>}, attribute_block = {data = 0x0,
   length = 0}, attribute_block_buffer = {data = 0x0, length = 0}, tile_known = {bits = 3888,
   vec = 0x805643800 ""}, rgb = 0x8064a1ea0, {server = {status = {vec = "\001"}, got_first_city = true,
     private_map = 0x80652c000, really_gives_vision = {vec = '\0' <repeats 15 times>}, debug = {
       vec = ""}, adv = 0x80543d800, ais = {0x80626f000, 0x0, 0x0},
     delegate_to = '\0' <repeats 47 times>, orig_username = '\0' <repeats 47 times>}, client = {
     tile_vision = {{bits = 257, vec = 0x80652c000 ""}, {bits = 0, vec = 0x0}}}}}
(gdb) p *pfrom
```

Figure 14. *In the pfrom pointer*

To print the variable's values, we just type 'p'. In this case 'p' is a pointer to a player structure. If we can check the definition of the player structure, we just type ptype pfrom and the structure definition will be displayed (Figure 13).

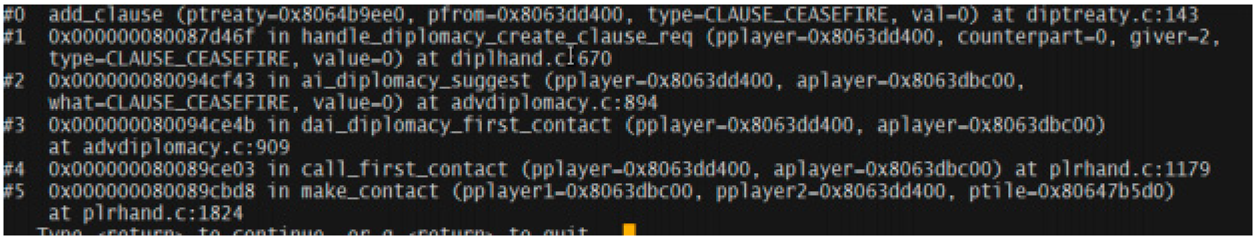
Now let's see what the values are for these fields for the demanding civilization. As the `pfrom` is a pointer we need to use pointer notation to check its contents (Figure 14).

And there we go the full dump for the player `struct` (Figure 15). Looking at the player `struct`, it seems that the leader name is Roy Jenkins and looking at the `backtrace (bt)`, the clause of the treaty seems to be "cease fire", so we are going to be offered a peace treaty (Figure 16). To continue executing the program type 'next' or 'n'; something like this will be displayed in the diplomacy tab (Figure 17).



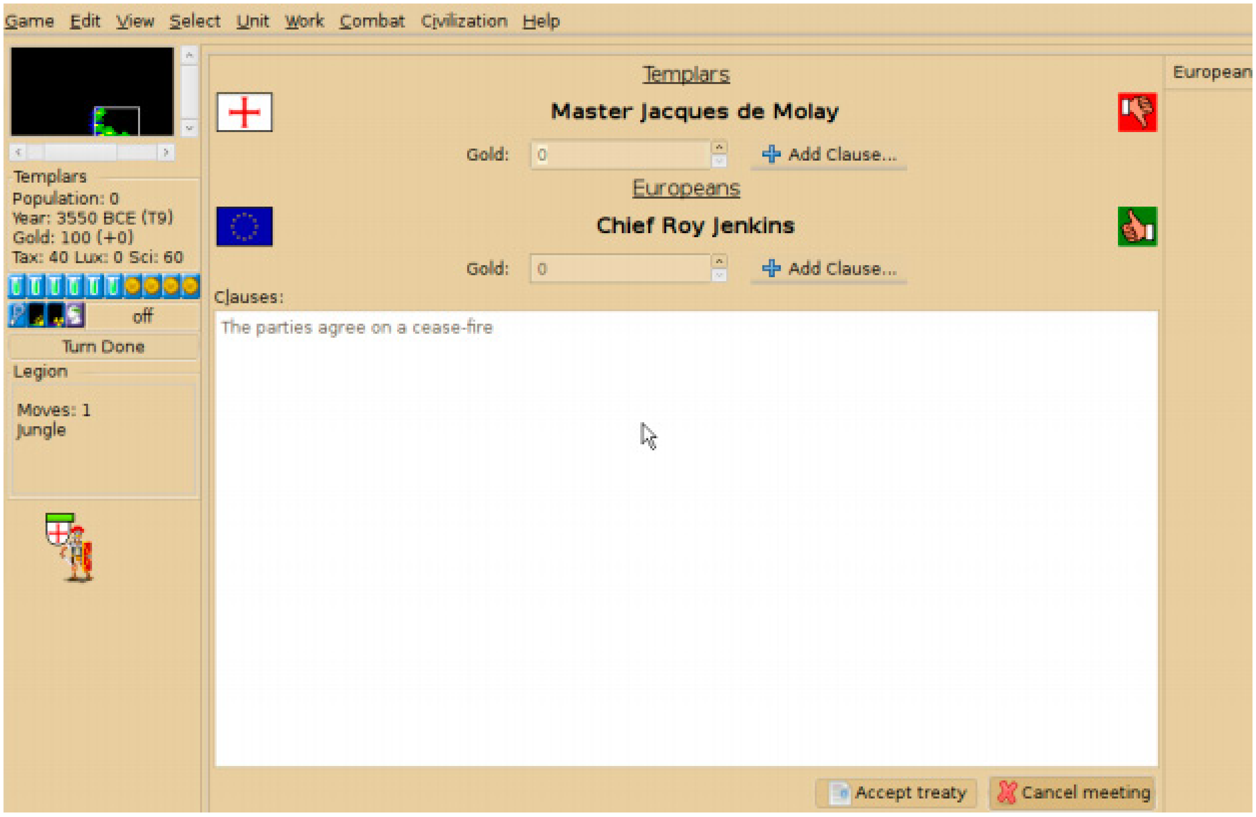
```
1 = {slot = 0x805407410, name = "Roy Jenkins", '\0' <repeats 36 times>,
 username = "Unassigned", '\0' <repeats 37 times>,
 ranked_username = "Unassigned", '\0' <repeats 37 times>, user_turns = 10, is_male = true,
 government = 0x805575500, target_government = 0x0, nation = 0x8068470c8, team = 0x805603fe0,
 is_ready = false, phase_done = false, nturns_idle = 9, is_alive = true, revolution_finishes = -1,
 real_embassy = {vec = '\0' <repeats 15 times>}, diplstates = 0x80540c400, city_style = 0,
 cities = 0x80682c3a0, units = 0x80682c3c0, score = {happy = 0, content = 2, unhappy = 0, angry = 0,
   specialists = {0 <repeats 20 times>}, wonders = 0, techs = 0, techout = 2, landarea = 35000,
   settledarea = 4000, population = 20, cities = 2, units = 0, pollution = 0, literacy = 0, bnp = 4,
   mfg = 5, spaceship = 0, units_built = 0, units_killed = 0, units_lost = 0, game = 2}, economic = {
--Type <return> to continue, or q <return> to quit---
```

Figure 15. *The full dump for the player struct*



```
#0  add_clause (ptreaty=0x8064b9ee0, pfrom=0x8063dd400, type=CLAUSE_CEASEFIRE, val=0) at diptreaty.c:143
#1  0x000000080087d46f in handle_diplomacy_create_clause_req (pplayer=0x8063dd400, counterpart=0, giver=2,
    type=CLAUSE_CEASEFIRE, value=0) at diplhand.c:670
#2  0x000000080094cf43 in ai_diplomacy_suggest (pplayer=0x8063dd400, aplayer=0x8063dbc00,
    what=CLAUSE_CEASEFIRE, value=0) at advdiplomacy.c:894
#3  0x000000080094ce4b in dai_diplomacy_first_contact (pplayer=0x8063dd400, aplayer=0x8063dbc00)
    at advdiplomacy.c:909
#4  0x000000080089ce03 in call_first_contact (pplayer=0x8063dd400, aplayer=0x8063dbc00) at plrhand.c:1179
#5  0x000000080089cbd8 in make_contact (pplayer1=0x8063dbc00, pplayer2=0x8063dd400, ptile=0x80647b5d0)
    at plrhand.c:1824
--Type <return> to continue, or q <return> to quit---
```

Figure 16. *A peace treaty*

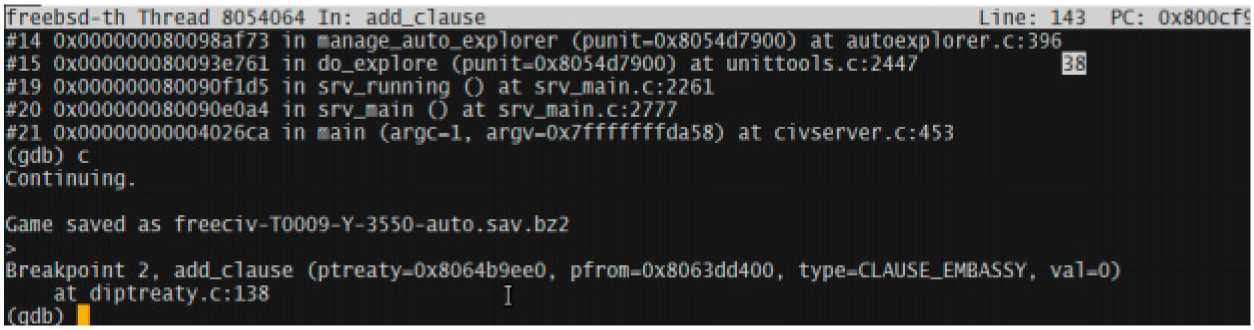


Templars
Master Jacques de Molay
Gold: 0
Europeans
Chief Roy Jenkins
Gold: 0
Clauses:
The parties agree on a cease-fire

Templars
Population: 0
Year: 3550 BCE (T9)
Gold: 100 (+0)
Tax: 40 Lux: 0 Sci: 60

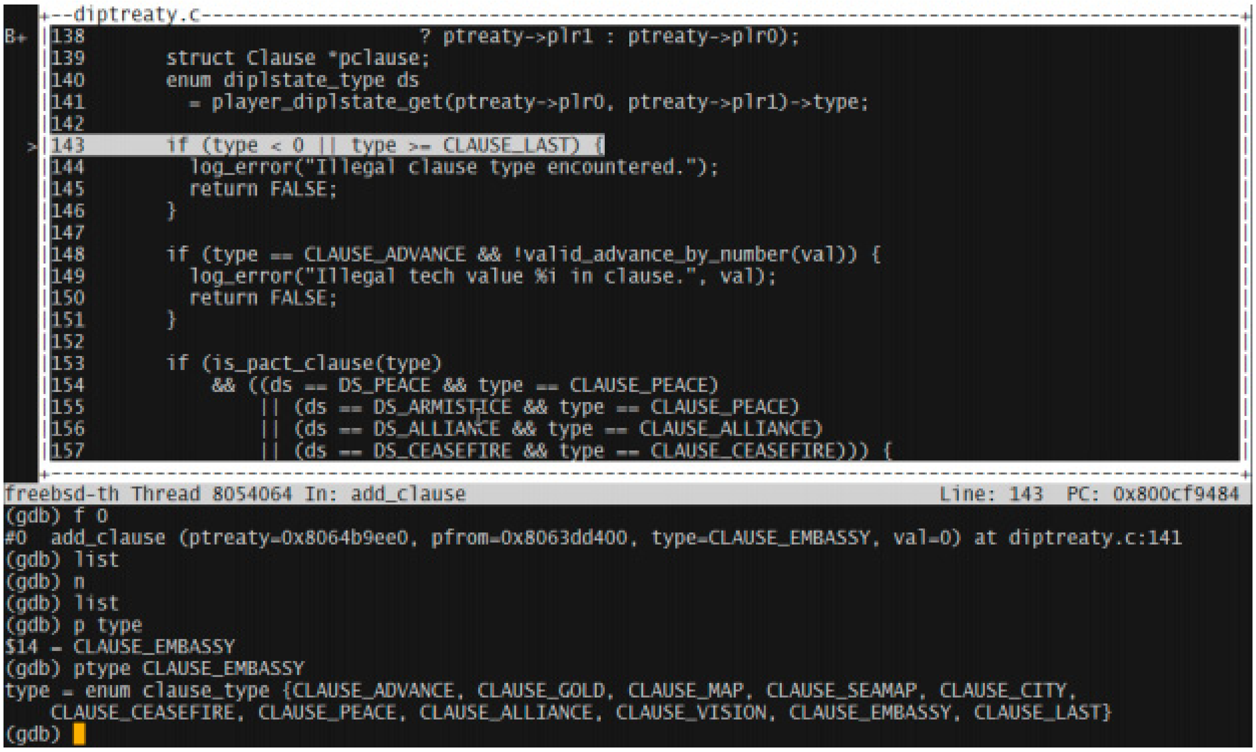
Legion
Moves: 1
Jungle

Figure 17. *In the diplomacy tab*



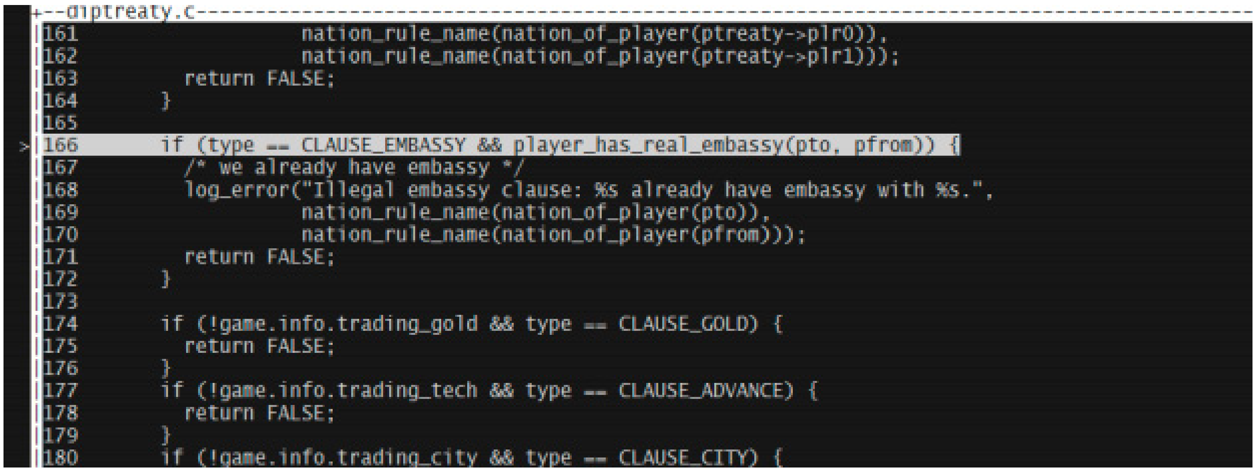
```
freebsd-th Thread 8054064 In: add_clause                                       Line: 143   PC: 0x800cf9
#14 0x000000080098af73 in manage_auto_explorer (punit=0x8054d7900) at autoexplorer.c:396
#15 0x000000080093e761 in do_explore (punit=0x8054d7900) at unittools.c:2447          38
#19 0x000000080090f1d5 in srv_running () at srv_main.c:2261
#20 0x000000080090e0a4 in srv_main () at srv_main.c:2777
#21 0x00000000004026ca in main (argc=1, argv=0x7fffffffda58) at civserver.c:453
(gdb) c
Continuing.

Game saved as freeciv-T0009-Y-3550-auto.sav.bz2
>
Breakpoint 2, add_clause (ptreaty=0x8064b9ee0, pfrom=0x8063dd400, type=CLAUSE_EMBASSY, val=0)
    at diptreaty.c:138
(gdb)
```

Figure 18. *The cease-fire treaty*



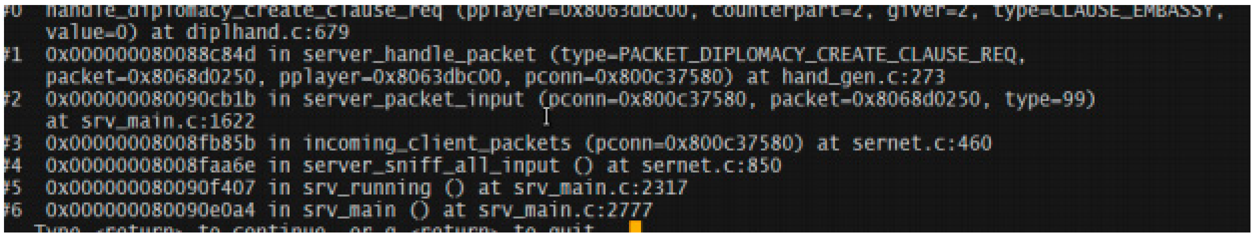
```
--diptreaty.c-----------------------------------------------------------------------------
                      ? ptreaty->plr1 : ptreaty->plr0);
        struct Clause *pclause;
        enum diplstate_type ds
          = player_diplstate_get(ptreaty->plr0, ptreaty->plr1)->type;

        if (type < 0 || type >= CLAUSE_LAST) {
          log_error("Illegal clause type encountered.");
          return FALSE;
        }

        if (type == CLAUSE_ADVANCE && !valid_advance_by_number(val)) {
          log_error("Illegal tech value %i in clause.", val);
          return FALSE;
        }

        if (is_pact_clause(type)
            && ((ds == DS_PEACE && type == CLAUSE_PEACE)
                || (ds == DS_ARMISTICE && type == CLAUSE_PEACE)
                || (ds == DS_ALLIANCE && type == CLAUSE_ALLIANCE)
                || (ds == DS_CEASEFIRE && type == CLAUSE_CEASEFIRE))) {
------------------------------------------------------------------------------------------
freebsd-th Thread 8054064 In: add_clause                                 Line: 143   PC: 0x800cf9484
(gdb) f 0
#0  add_clause (ptreaty=0x8064b9ee0, pfrom=0x8063dd400, type=CLAUSE_EMBASSY, val=0) at diptreaty.c:141
(gdb) list
(gdb) n
(gdb) list
(gdb) p type
$14 = CLAUSE_EMBASSY
(gdb) ptype CLAUSE_EMBASSY
type = enum clause_type {CLAUSE_ADVANCE, CLAUSE_GOLD, CLAUSE_MAP, CLAUSE_SEAMAP, CLAUSE_CITY,
    CLAUSE_CEASEFIRE, CLAUSE_PEACE, CLAUSE_ALLIANCE, CLAUSE_VISION, CLAUSE_EMBASSY, CLAUSE_LAST}
(gdb)
```

Figure 19. *The commands*



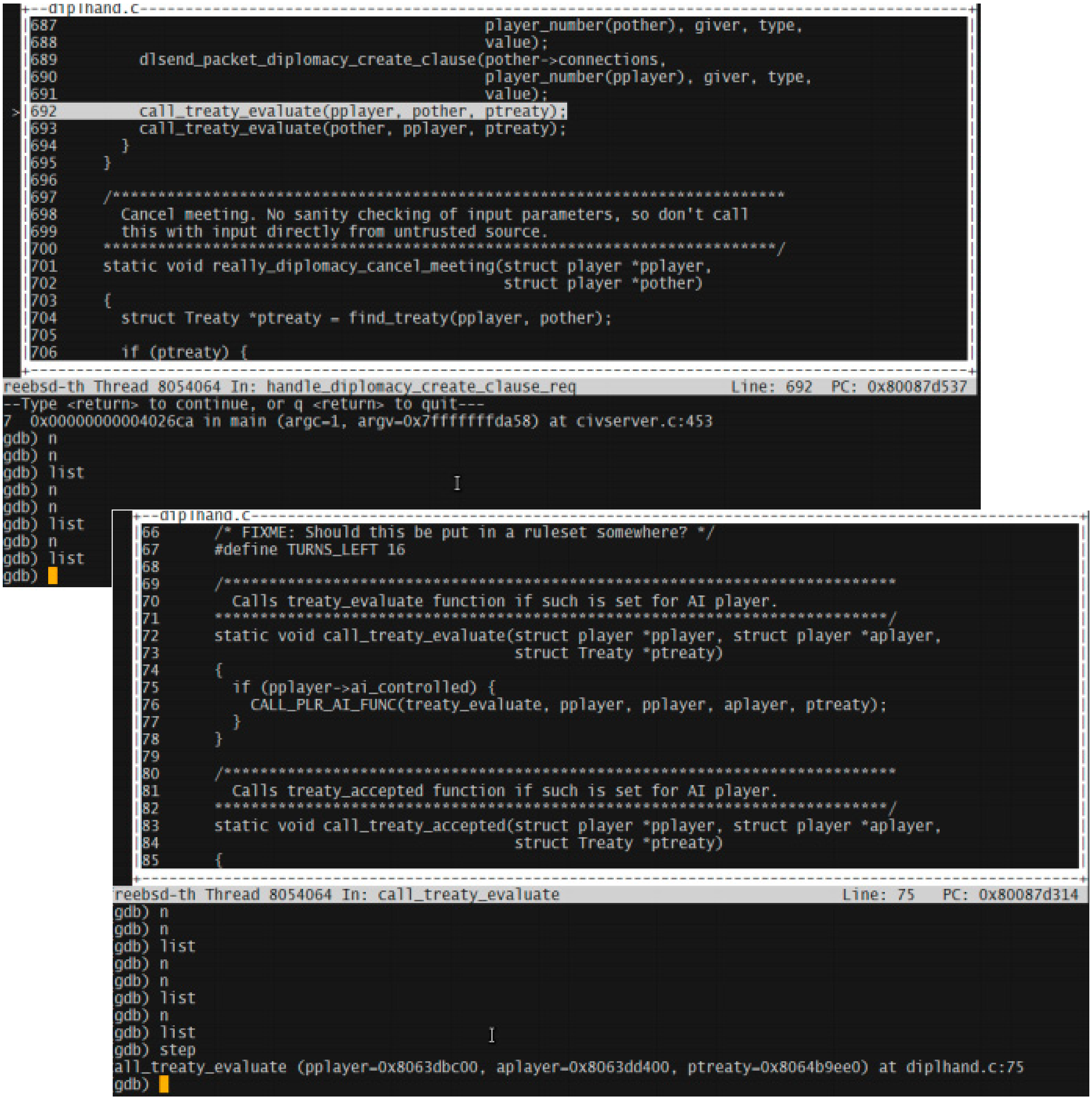
```
--diptreaty.c-----------------------------------------------------------------------------
                      nation_rule_name(nation_of_player(ptreaty->plr0)),
                      nation_rule_name(nation_of_player(ptreaty->plr1)));
          return FALSE;
        }

        if (type == CLAUSE_EMBASSY && player_has_real_embassy(pto, pfrom)) {
          /* we already have embassy */
          log_error("Illegal embassy clause: %s already have embassy with %s.",
                      nation_rule_name(nation_of_player(pto)),
                      nation_rule_name(nation_of_player(pfrom)));
          return FALSE;
        }

        if (!game.info.trading_gold && type == CLAUSE_GOLD) {
          return FALSE;
        }
        if (!game.info.trading_tech && type == CLAUSE_ADVANCE) {
          return FALSE;
        }
        if (!game.info.trading_city && type == CLAUSE_CITY) {
```

Figure 20. *The next*



```
#0  handle_diplomacy_create_clause_req (pplayer=0x8063dbc00, counterpart=2, giver=2, type=CLAUSE_EMBASSY,
    value=0) at diplhand.c:679
#1  0x000000080088c84d in server_handle_packet (type=PACKET_DIPLOMACY_CREATE_CLAUSE_REQ,
    packet=0x8068d0250, pplayer=0x8063dbc00, pconn=0x800c37580) at hand_gen.c:273
#2  0x000000080090cb1b in server_packet_input (pconn=0x800c37580, packet=0x8068d0250, type=99)
    at srv_main.c:1622
#3  0x00000008008fb85b in incoming_client_packets (pconn=0x800c37580) at sernet.c:460
#4  0x00000008008faa6e in server_sniff_all_input () at sernet.c:850
#5  0x000000080090f407 in srv_running () at srv_main.c:2317
#6  0x000000080090e0a4 in srv_main () at srv_main.c:2777
---Type <return> to continue, or q <return> to quit---
```

Figure 21. *The commands*



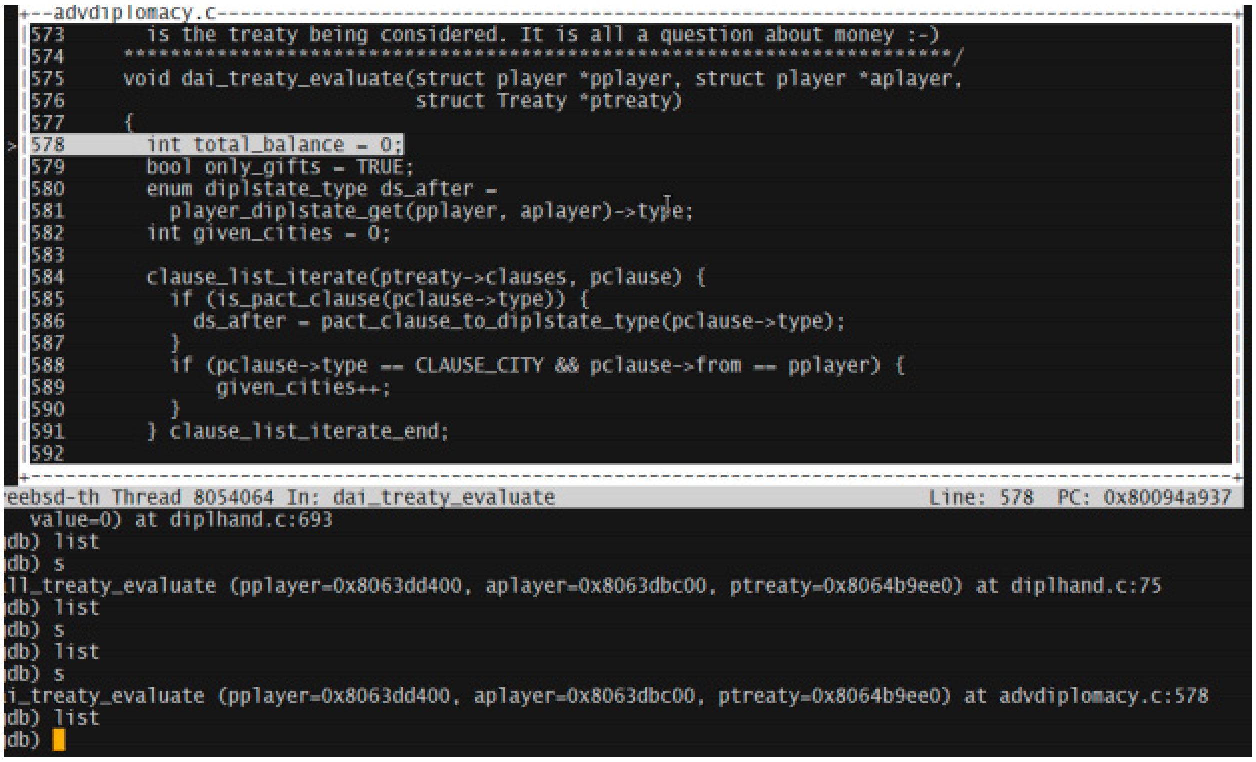
```
+--diplhand.c---------------------------------------------------------------------------------+
|687                                                   player_number(pother), giver, type,    |
|688                                                   value);                                |
|689           dlsend_packet_diplomacy_create_clause(pother->connections,                     |
|690                                                   player_number(pplayer), giver, type,   |
|691                                                   value);                                |
>|692          call_treaty_evaluate(pplayer, pother, ptreaty);                                 |
|693           call_treaty_evaluate(pother, pplayer, ptreaty);                                 |
|694         }                                                                                 |
|695       }                                                                                   |
|696                                                                                           |
|697       /**************************************************************************        |
|698         Cancel meeting. No sanity checking of input parameters, so don't call              |
|699         this with input directly from untrusted source.                                    |
|700       **************************************************************************/        |
|701       static void really_diplomacy_cancel_meeting(struct player *pplayer,                 |
|702                                                   struct player *pother)                  |
|703       {                                                                                   |
|704         struct Treaty *ptreaty = find_treaty(pplayer, pother);                            |
|705                                                                                           |
|706         if (ptreaty) {                                                                    |
+---------------------------------------------------------------------------------------------+
reebsd-th Thread 8054064 In: handle_diplomacy_create_clause_req          Line: 692   PC: 0x80087d537
--Type <return> to continue, or q <return> to quit---
7  0x00000000004026ca in main (argc=1, argv=0x7fffffffda58) at civserver.c:453
gdb) n
gdb) n
gdb) list
gdb) n
gdb) n
gdb) list
gdb) n
gdb) list
gdb) 
```

```
+--diplhand.c---------------------------------------------------------------------------------+
|66        /* FIXME: Should this be put in a ruleset somewhere? */                              |
|67        #define TURNS_LEFT 16                                                                |
|68                                                                                             |
|69        /**************************************************************************         |
|70          Calls treaty_evaluate function if such is set for AI player.                       |
|71        **************************************************************************/         |
|72        static void call_treaty_evaluate(struct player *pplayer, struct player *aplayer,     |
|73                                         struct Treaty *ptreaty)                             |
|74        {                                                                                    |
|75          if (pplayer->ai_controlled) {                                                      |
|76            CALL_PLR_AI_FUNC(treaty_evaluate, pplayer, pplayer, aplayer, ptreaty);           |
|77          }                                                                                  |
|78        }                                                                                    |
|79                                                                                             |
|80        /**************************************************************************         |
|81          Calls treaty_accepted function if such is set for AI player.                       |
|82        **************************************************************************/         |
|83        static void call_treaty_accepted(struct player *pplayer, struct player *aplayer,     |
|84                                         struct Treaty *ptreaty)                             |
|85        {                                                                                    |
+---------------------------------------------------------------------------------------------+
reebsd-th Thread 8054064 In: call_treaty_evaluate                        Line: 75    PC: 0x80087d314
gdb) n
gdb) n
gdb) list
gdb) n
gdb) n
gdb) list
gdb) n
gdb) list
gdb) step
all_treaty_evaluate (pplayer=0x8063dbc00, aplayer=0x8063dd400, ptreaty=0x8064b9ee0) at diplhand.c:75
gdb) 
```

Figure 22. *The commands*

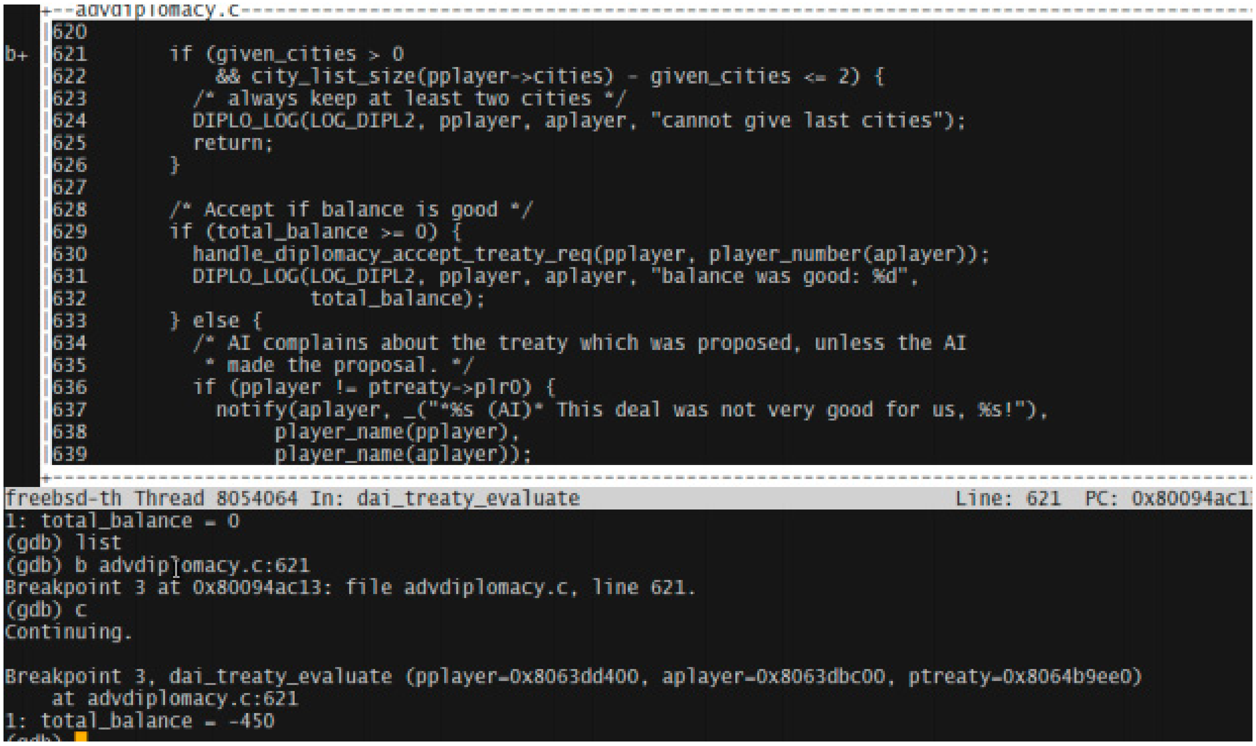
What you cannot see in the screenshot is that I have requested an embassy in return for the cease-fire treaty, but here, it is shown on Figure 18.

Let's go line by line using next; you could also use the step command but if you use the step command, it will take you inside a function call instead of just evaluating



```
+--advdiplomacy.c-------------------------------------------------------------------------+
 |573      is the treaty being considered. It is all a question about money :-)
 |574     **************************************************************************/
 |575     void dai_treaty_evaluate(struct player *pplayer, struct player *aplayer,
 |576                              struct Treaty *ptreaty)
 |577     {
>|578       int total_balance = 0;
 |579       bool only_gifts = TRUE;
 |580       enum diplstate_type ds_after =
 |581         player_diplstate_get(pplayer, aplayer)->type;
 |582       int given_cities = 0;
 |583
 |584       clause_list_iterate(ptreaty->clauses, pclause) {
 |585         if (is_pact_clause(pclause->type)) {
 |586           ds_after = pact_clause_to_diplstate_type(pclause->type);
 |587         }
 |588         if (pclause->type == CLAUSE_CITY && pclause->from == pplayer) {
 |589           given_cities++;
 |590         }
 |591       } clause_list_iterate_end;
 |592
+-----------------------------------------------------------------------------------------+
eebsd-th Thread 8054064 In: dai_treaty_evaluate                        Line: 578   PC: 0x80094a937
  value=0) at diplhand.c:693
db) list
db) s
ll_treaty_evaluate (pplayer=0x8063dd400, aplayer=0x8063dbc00, ptreaty=0x8064b9ee0) at diplhand.c:75
db) list
db) s
db) list
db) s
i_treaty_evaluate (pplayer=0x8063dd400, aplayer=0x8063dbc00, ptreaty=0x8064b9ee0) at advdiplomacy.c:578
db) list
db)
```

Figure 23. *The following commands*



```
+--advdiplomacy.c-----------------------------------------------------------------------
   |620
b+ |621       if (given_cities > 0
   |622           && city_list_size(pplayer->cities) - given_cities <= 2) {
   |623         /* always keep at least two cities */
   |624         DIPLO_LOG(LOG_DIPL2, pplayer, aplayer, "cannot give last cities");
   |625         return;
   |626       }
   |627
   |628       /* Accept if balance is good */
   |629       if (total_balance >= 0) {
   |630         handle_diplomacy_accept_treaty_req(pplayer, player_number(aplayer));
   |631         DIPLO_LOG(LOG_DIPL2, pplayer, aplayer, "balance was good: %d",
   |632                   total_balance);
   |633       } else {
   |634         /* AI complains about the treaty which was proposed, unless the AI
   |635          * made the proposal. */
   |636         if (pplayer != ptreaty->plr0) {
   |637           notify(aplayer, _("*%s (AI)* This deal was not very good for us, %s!"),
   |638                  player_name(pplayer),
   |639                  player_name(aplayer));
+---------------------------------------------------------------------------------------
freebsd-th Thread 8054064 In: dai_treaty_evaluate                       Line: 621   PC: 0x80094ac1
1: total_balance = 0
(gdb) list
(gdb) b advdiplomacy.c:621
Breakpoint 3 at 0x80094ac13: file advdiplomacy.c, line 621.
(gdb) c
Continuing.

Breakpoint 3, dai_treaty_evaluate (pplayer=0x8063dd400, aplayer=0x8063dbc00, ptreaty=0x8064b9ee0)
    at advdiplomacy.c:621
1: total_balance = -450
(gdb)
```

Figure 24. *The commands for program execution*

the `function` and returning like the next command (Figure 19).

We are currently at line 143; I just checked what kind of data type was CLAUSE_EMBASSY, it was a enum one (somewhat obvious). Using next a couple of times will get us to the next step. See Figure 20.

Keep on typing 'n' and we will exit from the function call and arrive to `handle_diplomacy_create_clause_req` (Figure 21).

Let's keep on typing 'next' and we will arrive to this function `call_treaty_evaluate`. That seems interesting. Maybe the results of rejection or acceptance of conditions are done. As I explained earlier, we can step into this one using the step command (Figure 22).

Let's step all the way to get to another point in the program execution; after a couple of steps are shown on Figure 23.

So a quick glance at the source code tells us that the total_balance variable is somewhat important to evaluate if a clause is accepted (In our case we are requesting to give us an embassy). Instead of printing this variable multiple times, let's leave it available in the display.

```
#(gdb) display total_balance
```

Then we set a breakpoint somewhere ahead of `advdiplomacy.c:621`; we can see that the `total _ balance` value is displayed and it is -450 (seems bad for our proposal).

As we can see, `total_balance >=0` is the condition to approve the proposal. This is a review of the commands used in this session:

COMMAND	ABBREVIATED FORM	WHAT IT DOES
info local	i lo	Print values and names of all local variables in the current scope.
backtrace	bt	A backtrare is a summary of how your program got where it is. It shows one line per fatale, for many frames, starting with t e currently executing frame (frame zero), followed by its caller (frame one), end on up the stock.
frame <frame number>	f <frame number>	The call stack is divided up into contiguous pieces called stack frames, or frames for short; each frame is the data associated with one call to one function. The frame contains the arguments given to the function, the function 's local variables, and address at which the function is executing.
print <variable>	p <variable>	displays the value of the variable
display <variable>	disp <variable>	Will automatically print the value the variable being displayed as long as it is within the scope
win	Win	Will enter gdb in tui (text user interface) mode if we did not entered in the first place. Default layout is source at the top commands at the bottom.
next	n n <number of next to perform>	Execute next line of code. Will not enter functions. You can use as parameter the number or times to execute next
step	s s<number of step to Perform>	Step to next line of code. Will step into a function.

These are really basic commands, but really useful.

Advanced inspection of data structures and variables

Now that we have used the display command or the print command, it is getting pretty tedious to manually inspect a variable or data structure by typing p or display every time we hit a breakpoint we have set. There is a command called commands to save us from all this typing.

First we set a breakpoint where we want to automatically inspect data. In this case I'll check one of the city functions.

```
(gdb) b city.c:2352
(gdb) 4 breakpoint keep y 0x0000000800cf1b7b in citizen_
   base_mood at
city.c:2352
```

Now we can type the following: `commands <breakpoint number>`

```
(gdb) commands 4
```

Type commands for when breakpoint 4 is hit, one per line.
End with a line saying just "end".

```
>
```

After you have set the instructions to be executed after the breakpoint is hit, you could modify them or just erase them like this:

```
(gdb) commands 4
```

Type commands for when breakpoint 4 is hit, one per line.
End with a line saying just "end".

```
> end
```

Now if you want to execute something:

```
(gdb) commands 4
```

Type commands for when breakpoint 4 is hit, one per line.
End with a line saying just "end".> printf "Setting city mood for leader: %s", pplayer->name

```
> end
```

Now we can type all the instructions we want to be executed when this breakpoint is hit. Usually, we use print to display values, but there is a more powerful function called printf that uses a similar format as the C-language function:

```
(gdb) printf "%s", pplayer->name
```

As in C printf, ordinary characters in the template are printed verbatim, while conversion specification introduced by the '%' character causes subsequent expressions to be evaluated, their values converted and formatted according to type and style information encoded in the conversion specifications, and then printed.

For example, you can print two values in hex like this:

```
printf "foo, bar-foo = 0x%x, 0x%x\n", foo, bar-foo
```

printf supports all the standard C conversion specifications, including the flags and modifiers between the '%' character and the conversion letter, with the following exceptions:

The argument-ordering modifiers, such as '2$', are not supported.

The modifier '*' is not supported for specifying precision or width.

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

Multi-talented, long-haired, bearded, Open Source IT guru seeks employment sanctuary with ethical organisation.

With over 40 years technology exposure as both an amateur and a professional, I migrated to the IT sector from an analogue electronics background and have worked in a variety of senior technical roles from Senior Support Engineer, System Administrator, Developer and Architect to Group IT manager. My core speciality is developing and commissioning innovative Linux and *BSD solutions as well as identifying the best ways to leverage technology to fit the culture of an organisation. I also have a keen passion for writing, and am a regular contributor to BSD magazine.

My career has stretched across multiple sectors from local and national government through entrepreneurial SME's, blue chips (including airlines, automotive, PR / media, oil-field services) and freelance consultancy. In my current role as a systems engineer I have successfully advocated and implemented Open Source solutions in an organisation previously moribund by proprietary and closed source products, generating substantial cost savings as a result.

The most effective teams are built on strong, communicative, interactive relationships.

Paradoxically, I combine a "hard" technological mindset with strong "soft" people skills. A Myers-Briggs type INFJ, it is essential for me that any culture I serve in embraces integrity, creativity, humanity and ethics. Ergo, quality is not something you believe in, quality is something you experience.

My IT career and professional philosophy has been shaped by the late Michael Kidron, author of "The State Of The World Atlas", and Béla Hatvany, founder of Silver Platter Information Ltd, who I am both deeply indebted to for hiring me in my first IT role. If you are looking for a truly different type of IT guru, please get in touch.

linuxgreybeard@gmail.com

The ‘'’ flag (for separation of digits into groups according to LC_NUMERIC’) is not supported.

The type modifiers ‘hh’, ‘j’, ‘t’, and ‘z’ are not supported.

The conversion letter ‘n’ (as in ‘%n’) is not supported.

The conversion letters ‘a’ and ‘A’ are not supported.

Note that the ‘ll’ type modifier is supported only if the underlying C implementation used to build GDB supports the long long int type, and the ‘L’ type modifier is supported only if long double type is available.

As in C, printf supports simple backslash-escape sequences, such as \n, ‘\t’, ‘\\’, ‘\"’, ‘\a’, and ‘\f’, that consist of backslash followed by a single character. Octal and hexadecimal escape sequences are not supported.

Additionally, printf supports conversion specifications for DFP (Decimal Floating Point) types using the following length modifiers together with a floating point specifier. Letters: ‘H’ for printing Decimal32 types.

‘D’ for printing Decimal64 types. ‘DD’ for printing Decimal128 types.

If the underlying C implementation used to build GDB has support for the three length modifiers for DFP types, other modifiers such as width and precision will also be available for GDB to use.

In case there is no such C support, no additional modifiers will be available and the value will be printed in the standard way. Here’s an example of printing DFP types using the above conversion letters:

```
printf "D32: %Hf - D64: %Df - D128: %DDf\
   n",1.2345df,1.2E10dd,1.2E1dl
```

Dynamically allocated arrays

Sometimes. it’s better to put most of the type we will need to take a look at in the contents of dynamically allocated arrays (the ones created by malloc and calloc system calls).

For example we have the usual static memory array:

```
char t[8001];
```

It’s easy to display its contents using

```
(gdb) p t
```

But what about this one:

```
int *t; …
t = (int *) malloc ( 8001 * sizeof( int) );
(gdb) p t
```

This will give only the address

```
(gdb) p *t
```

This will give you the data of the first element in the array, so what is the solution?

```
(gdb) p *t@25
```

This command will print 25 elements from the array t; the format is pointer@<number of elements.

Getting information from the symbol table

When we compiled our program with the –g flag, we instructed the compiler to generate a symbol table in our program binary. This table contains variable names, function names and types. Now let’s suppose we want to know the names of all the functions available. We could use one of the info family commands:

```
(gdb) info functions
```

This command will print the names and data types of all defined functions. If we want to check only the function names matching a regexp we use the command: info functions `<regexp>`.

For example:

```
(gdb) info functions city
```

Will match all functions that have city string in their name, you must use grep regexp not perl’s regexp.

The same goes with variables with the command:

```
(gdb) info variables
```

Print the names and data types of all variables that are declared outside of functions (not the local variables).

Also the same syntax for info variables regexp (gdb) info variables city. Print the names and data types of all variables (except for local variables) whose names contain a match for regular expression regexp.

```
(gdb) info address symbol
```

Describe where the data for the symbol is stored. For a register variable, this says which register it is kept in. For a non-register local variable, this prints the stack-frame offset at which the variable is always stored. Note the contrast with ‘print &symbol’, which does not work at all for a register variable, and for a stack local variable prints the exact address of the current instantiation of the variable.

```
(gdb) whatis exp
```

Print the data type of expression exp. exp is not actually evaluated, and any side-effecting operations (such as assignments or function calls) inside it do not take place. Any kind of constant, variable or operator defined by the programming language you are using is valid in an expression in GDB.

```
(gdb) whatis
```

Print the data type of $, the last value in the value history.

```
(gdb) ptype typename
```

Print a description of data type typename. typename may be the name of a type, or for C code it may have the form 'class class-name', 'struct struct-tag', 'union union-tag' or 'enum enum-tag'.

```
(gdb) ptype exp
ptype
```

Print a description of the type of expression exp. ptype differs from whatis by printing a detailed description, instead of just the name of the type. For example, for this variable declaration:

```
struct example {double dtype; float ftype} ex1;
```

The two commands give this output:

```
(gdb) whatis ex1
type = struct example
(gdb) ptype ex1
type = struct example {
double dtype;
float ftype;
}
```

As with what is, using ptype without an argument refers to the type of $, the last value in the value history.

```
(gdb) info types regexp
```

Print a brief description of all types whose name matches regexp (or all types in your program, if you supply no argument). Each complete typename is matched as though it were a complete line; thus, 'i type value' gives information on all types in your program whose name includes the string value, but 'i type ^value$' gives information only on types whose complete name is value.

This command differs from ptype in two ways: first, like whatis, it does not print a detailed description; second, it lists all source files where a type is defined.

```
(gdb) info source
```

Show the name of the current source file–that is, the source file for the function containing the current point of execution–and the language it was written in. (gdb) info sources.

Print the names of all source files in your program for which there is debugging information, organized into two lists: files whose symbols have already been read, and files whose symbols will be read when needed.

```
(gdb) info functions
```

Print the names and data types of all defined functions.

```
(gdb) info functions regexp
```

Print the names and data types of all defined functions whose names contain a match for regular expression regexp. Thus, 'info fun step' finds all functions whose names include step; 'info fun ^step' finds those whose names start with step.

```
(gdb) info variables
```

Print the names and data types of all variables that are declared outside of functions (i.e., excluding local variables).

```
(gdb) info variables regexp
```

Print the names and data types of all variables (except for local variables) whose names contain a match for regular expression regexp.

Conclusions

In GDB we have three ways of interrupting the program flow and inspecting what we need; breakpoints, watchpoints and catchpoints.

A breakpoint stops the execution at a particular location within the program. We have temporary breakpoints, regexp breakpoints and we could set conditional breakpoints.

The usual breakpoint :

```
(gdb) break <source>:<line>
(gdb) break <source.c>:<function>
(gdb) break 3 This one stops at line 3 of the current
```

```
   source file being executed.
(gdb) break <function>
```

The temporary break point is a simple breakpoint that is deleted after it is hit; the command for this is:

```
(gdb) tbreak <same format as breakpoint>
```

The regexp breakpoint sets breakpoints at the functions matching the regexp provided

```
(gdb) rbreak ^cityConditional breakpoint, stops the
   execution of the program only if the condition is met
(gdb) b if strcmp(commands[0].synopsis,"*start") ==0
```

Yes, you could use the C library functions as long as your program is linked against libc.

You can enable or disable breakpoints with the following command:

```
enable once — Enable breakpoints for one hit
enable delete — Enable breakpoints and delete when hit
(gdb) enable once 1
(gdb) enable delete 1
```

A watchpoint stops the execution when a particular memory location (or an expression involving one or more locations) changes value. Depending on your system, watchpoints may be implemented in software or hardware. GDB does software watchpointing by single-stepping your program and testing the variable's value each time, which is hundreds of times slower than normal execution, but it's really useful if you really don't have a clue of where the problem is in your program.

The syntax for this command is: watch <expr>

```
(gdb) watch commands[0]
Watchpoint 1: commands[0]
```

A catchpoint stops the execution when a particular event occurs. The event could be one of the following.

Raised signals may be caught:

```
catch signal - all signals
catch signal <signame> - a particular signal
```

Raised exceptions may be caught:

- catch throw – all exceptions, when thrown
- catch throw <exceptname> – a particular exception, when thrown
- catch catch – all exceptions, when caught
- catch catch <exceptname> – a particular exception, when caught

Thread or process events may be caught:

- catch thread_start – any threads, just after creation
- catch thread_exit – any threads, just before expiration
- catch thread_join – any threads, just after joins

Process events may be caught:

- catch start – any processes, just after creation
- catch exit – any processes, just before expiration
- catch fork – calls to `fork()`
- catch vfork – calls to `vfork()`
- catch exec – calls to `exec()`

Dynamically-linked library events may be caught:

- catch load – loads of any library
- catch load <libname> – loads of a particular library
- catch unload – unloads of any library
- catch unload <libname> – unloads of a particular library

The act of your program's execution stopping may also be caught:

- catch stop
- C++ exceptions may be caught:
- catch throw – all exceptions, when thrown
- catch catch – all exceptions, when caught

You can enable and delete breakpoints, watchpoints and catchpoints with the enable and delete command.

ABOUT THE AUTHOR

Carlos Neira has worked several years as a C/C++ developer and kernel porting and debugging enterprise legacy applications. He is currently employed as a C developer under Z/OS, debugging and troubleshooting legacy applications for a global financial company. Also he is engaged in independent research on affective computing. In his free time he contributes to the PC-BSD project and enjoys metal detecting.

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

A Complete Guide to FreeNAS Hardware Design,

Part I: Purpose and Best Practices

JOSH PAETZEL

A guide to selecting and building FreeNAS hardware, written by the FreeNAS Team, is long past overdue by now. For that, we apologize. The issue was the depth and complexity of the subject, as you'll see by the extensive nature of this four part guide, due to the variety of ways FreeNAS can be utilized. There is no "one-size-fits-all" hardware recipe. Instead, there is a wealth of hardware available, with various levels of compatibility with FreeNAS, and there are many things to take into account beyond the basic components, from use case and application to performance, reliability, redundancy, capacity, budget, need for support, etc. This document draws on years of experience with FreeNAS, ZFS, and the OS that lives underneath FreeNAS, FreeBSD. Its purpose is to give guidance on intelligently selecting hardware for use with the FreeNAS storage operating system, taking the complexity of its myriad uses into account, as well as providing some insight into both pathological and optimal configurations for ZFS and FreeNAS.

A word about software defined storage

FreeNAS is an implementation of Software Defined Storage; although software and hardware are both required to create a functional system, they are decoupled from one another. We develop and provide the software and leave the hardware selection to the user. Implied in this model is the fact that there are a lot of moving pieces in a storage device (figuratively, not literally). Although these parts are all supposed to work together, the reality is that all parts have firmware, many devices require drivers, and the potential for there to be subtle (or gross) incompatibilities is always present.

Best Practices

ECC RAM or Not?

This is probably the most contested issue surrounding ZFS (the filesystem that FreeNAS uses to store your data) today. I've run ZFS with ECC RAM and I've run it without. I've been involved in the FreeNAS community for many years and have seen people argue that ECC is required and others argue that it is a pointless waste of money. ZFS does something no other filesystem you'll have available to you does: it checksums your data, and it checksums the metadata used by ZFS, and it checksums the checksums. If your data is corrupted in memory before it is written, ZFS will happily write (and checksum) the corrupted data. Additionally, ZFS has no pre-mount consistency checker or tool that can repair filesystem damage. This is very nice when dealing with large storage arrays as a 64TB pool can be mounted in seconds, even after a bad shutdown. However if a non-ECC memory module goes haywire, it can cause irreparable damage to your ZFS pool that can cause complete loss of the storage. For this reason, I **highly** recommend the use of ECC RAM with "mission-critical" ZFS. Systems with ECC RAM will correct single bit errors on the fly, and will halt the system before they can do any damage to the array if multiple bit errors are detected. If it's imperative that your ZFS based system *must always* be available, ECC RAM is a requirement. If it's only some level of annoying (slightly, moderately…) that you need to restore your ZFS system from backups, non-ECC RAM will fit the bill.

How Much RAM is needed?

FreeNAS requires 8 GB of RAM for the base configuration. If you are using plugins and/or jails, 12GB is a better starting point. There's a lot of advice about how RAM hungry ZFS is, how it requires massive amounts of RAM, an oft quoted number is 1GB RAM per TB of storage. The reality is, it's complicated. ZFS does require a base level of RAM to be stable, and the amount of RAM it needs to be stable does grow with the size of the storage. 8GB of RAM will get you through the 24TB range. Beyond that 16GB is a safer minimum, and once you get past 100TB of storage, 32GB is recommended. However, that's just to satisfy the stability side of things. ZFS performance lives and dies by its caching. There are no good guidelines for how much cache a given storage size with a given number of simultaneous users will need. You can have a 2TB array with 3 users that needs 1GB of cache, and a 500TB array with 50 users that need 8GB of cache. Neither of those scenarios are likely, but they are possible. The optimal cache size for an array tends to increase with the size of the array, but outside of that guidance, the only thing we can recommend is to *measure and observe* as you go. FreeNAS includes tools in the GUI and the command line to see cache utilization. If your cache hit ratio is below 90%, you will see performance improvements by adding cache to the system in the form of RAM or SSD L2ARC (dedicated read cache devices in the pool).

RAID vs. Host Bus Adapters (HBAs)

ZFS wants direct control of the underlying storage that it is putting your data on. Nothing will make ZFS more unstable than something manipulating bits underneath ZFS. Therefore, connecting your drives to an HBA or directly to the ports on the motherboard is preferable to using a RAID controller; fortunately, HBAs are cheaper than RAID controllers to boot! If you must use a RAID controller, disable all write caching on it and disable all consistency checks. If the RAID controller has a passthrough or JBOD mode, use it. RAID controllers will complicate disk replacement and improperly configuring them can jeopardize the integrity of your volume (Using the write cache on a RAID controller is an almost sure-fire way to cause data loss with ZFS, to the tune of losing the entire pool).

Virtualization vs. Bare Metal

FreeBSD (the underlying OS of FreeNAS) is not the best virtualization guest: it lacks some virtio drivers, it lacks some OS features that make it a better behaved guest, and most importantly, it lacks full support from some virtualization vendors. In addition, ZFS wants direct access to your storage hardware. Many virtualization solutions only support hardware RAID locally (I'm looking at you, VMware) thus leading to enabling a worst case scenario of passing through a virtual disk on a datastore backed by a hardware RAID controller to a VM running FreeNAS. This puts two layers between ZFS and your data, one for the Host Virtualization's filesystem on the datastore and another on the RAID controller. If you can do PCI passthrough of an HBA to a FreeNAS VM, and get all the moving pieces to work properly, you can successfully virtualize FreeNAS. We even include the guest VM tools in FreeNAS for VMware, mainly because we use VMware to do a lot of FreeNAS development. However if you have problems, there are no developer assets running FreeNAS as a production VM and help will be hard to come by. For this reason, I highly recommend that FreeNAS be run "On the Metal" as the only OS on dedicated hardware.

Has the technology sector finally slid into the realm of used car salesmen, lawyers and ambulance chasers?

ROB SOMERVILLE

Adobe Photoshop inventor Thomas Knoll has made a call for the ethical use of the product. Has the technology sector finally slid into the realm of used car salesmen, lawyers and ambulance chasers?

NSA, security and social media ethics aside, there has been a discernible slide from grace over the years as to how IT departments, support staff etc. have been perceived both by society and management. While technology itself may be a factor in this, my personal theory is not that technologists are any less moral *per se* than any other professional sector, but rather when people fear what they do not understand there is an instinctive impulse to demonise, categorise, pigeon-hole and control. Culturally, we may laugh at the classifications of "Geek" and "Nerd", but ultimately outside of the technical community, these are terms of insult rather than endearment – designed to put the target firmly in their place. None is a more hypocritical sight than an HR department spewing forth never-ending drivel concerning equality, diversity and fairness yet at the same time categorising staff in IT as "washing machine engineers". Apparently, IT is now such a demystified specialisation, anyone with a screwdriver, a box cutter and a hammer (What – you don't use hammers when upgrading a server?) is sufficiently competent. Sadly, this devaluation of skills and value has permeated management – both middle and senior.

Maybe it is my non-cynical side coming out, but I don't believe for a moment that money alone is the main driver for career or indeed personal satisfaction for the majority of people. I would continue to work in IT even I wasn't paid – provided my family and myself had a roof over our heads and food in our belly. The lie that we must be continual consumers has finally died along with the myth that bankers have integrity, poverty is good for the soul and that the Western world we live in is a democracy. What drives people is a social contract, that they add value by the works of their hands and in return from their employer, get little bits of paper or lumps of metal that they can exchange for whatever they want. The old model stated that we might not give you so many tokens now, but we will give you a decent pension, flexible working conditions, and maybe other benefits such as more holidays per year and job stability. This fitted well with a socially co-operative model, where the essential servants of society (e.g. Law enforcement, Nurses, Civil servants etc.) found a good deal of job satisfaction. If you want more benefits, move to a more corporately aggressive sector, but in turn you will be expected to compromise your professional integrity more, perform a job you hate, spend less time with your family, or just do something morally repugnant and hope that your conscience doesn't hurt too much. And of course, all with the added benefit of little or no job security. The downward spiral of rising costs, increased competition and the decreased buying power of salaries is finally hitting home across the professional, skilled and semi-skilled marketplace. The old model is dead. The new model demands the commercial stance of using technology to cut costs and bring efficiencies and it brings with it a very two edged sword. The life of a well-designed enterprise scale system can be measured in decades, and once the developers, designers, analysts and programmers have left, provided the business model of the corporate entity does not change much, the return on investment can be huge and the creatives and others scaled down. So, in parallel with the false perception that anyone can do IT well now that it has become commoditised, we are heading towards another industrial revolution driven by excesses of the military industrial complex. Far from learning from the past and using technology to bring ethical, moral and social benefit, corporate culture has bitten the very hand that feeds it, turned, and started to consume its own children. To hell with the human and moral consequences, profit and efficiency is all that counts and it matters not how many people are put on the scrapheap in the process – be they skilled or unskilled. And we wonder why the alarm bells are going off by Bill Gates, Stephen Hawking and Elon Musk about the dangers of Artificial Intelligence.

I don't believe that any creation can be greater than it's creator. That does not mean that any creation in the hands of men cannot pose a significant threat to society. In the old days, programmers regularly put back doors in their software for maintenance purposes. The kill switch was always there. The danger is when we get to the stage that neither the end user nor the designer can effectively hit the kill switch and this authority is delegated to "the system" – and by that I mean either cultural, political or a hybrid mix of hardware and software. Case in point, we take the mobile phone network for granted but in a time of local emergency this can be dedicated to Law enforcement etc. cutting off the average consumer. While I don't have a problem with this particular scenario itself, what would happen if this control was exploited for political or economic purposes? Indeed this is happening now, but not by the technologists. The HR departments first port of call when hiring is Facebook, Twitter or LinkedIn. All well and good you might think, but what about decisions being made about your credit worthiness based on your associates or lifestyle? Your insurance company? The robotic trawling of these sites is a feature of the corporate landscape, from HR to marketing and reputation management. It is not the technology or the technologists, it is the power behind the throne, the men in grey suits, the hand behind the curtain that truly drives the agenda – and those unaccountable faceless corporate clones that either wittingly or unwittingly go along with the agenda.

The irony is that most IT professionals I have encountered are a decent bunch whose heart is to improve things and make life better, more interesting, more fun. Unfortunately, we tend to be mesmerised by the environment we work in, and forget that outside the IT suite there are those – unlike the machines and systems we work with – who have rather dark and ulterior motives. As our culture and society becomes more and more globalised – yet ironically at the same time more compartmentalised – it is difficult to see this as the layers of management and ethical responsibility are diluted. Nevertheless, by association, we are being tarred with the same brush. Too often, the politicians, leaders and the establishment have echoed the benefits of change, automation, technology and progress but the end result has been far from the idyllic. For the UK, in the 80's we moved to a service rather than a manufacturing based economy. 35 years later there are still areas that are blighted by unemployment, especially amongst the youth. Be it the Chicago motor industry, shipbuilding in Glasgow or farming in France, the economic winds of change blow, but there remains a fundamental disconnect between the vision, the implementation and the consequences.

As technologists we must come to face the harsh truth that we fall into the same category as gunsmiths. What we design is not bad in itself, it is how it is utilised that matters. Sadly, more than ever we better make sure that as a profession we are not exploited as useful idiots by those that wish to use our creative talents for evil and not for good. We urgently need to embrace a strong moral ethic. For as history has shown, should we experience a revolution on the scale that Gates, Hawking and Musk envisage, the creatives, intelligentsia and the useful idiots will be the first to face the wrath.

ABOUT THE AUTHOR

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.

Information Security Analytics

Finding Security Insights, Patterns, and Anomalies in Big Data. Simulations and Security Processes

MARK RYAN TALABIS, ROBERT MCPHERSON, INEZ MIYAMOTO AND JASON L. MARTIN

The primary tool that we will be using in this chapter about simulations is Arena, which is commercial software developed by Rockwell Automation. Arena is a powerful modeling and simulation software allowing a user to model and run simulation experiments. We will be using a fully functioning perpetual evaluation version, which is available for study and download at (http://www.arenasimulation.com/Tools_Resources_Download_Arena.aspx).

Let us get started with simulations. Since Arena is a Windows desktop application, when you start using the program, you will see three regions on the main Arena window. Let us familiarize you with the three regions:

- At the left-hand side of the main window, you will find the Project bar containing three tabs: basic process, report and navigate panel. In the Project bar, you will also find various "Arena modules" to be used when building a simulation model. We will discuss more about Arena modules in the latter part of this section.
- At the right-hand side, you will find the Model window flowchart view to be the largest part of your screen because it is your workspace where you will create models. You will be creating graphical models using flowcharts, images, animations, and other drawn elements.
- At the bottom part of the flowchart view, you will find the Model window spreadsheet view, which presents all the data associated with the model.

This chapter will provide a high-level overview of creating simulations in Arena. There are three main steps to making a simulation in Arena:

- Design and create a model,
- Add data and parameters to a model,
- Run a simulation, and
- Analyze a simulation.

Designing and Creating a Model

Before we start in Arena, we first need to create a "conceptual model" for a scenario we will simulate. A conceptual model is how you think a process should work – this could be anything from you just drawing it out on a piece of paper or just thinking about it.

Once you have a conceptual model, the next step is to build the model in the workspace using the "modules" in Arena. Modules are the building blocks of a model. There are two kinds of modules: the flowchart modules and the data modules.

Information Security Analytics. http://dx.doi.org/10.1016/B978-0-12-800207-0.00004-6
Copyright © 2015 Elsevier Inc. All rights reserved.

The flowchart modules illustrate the logic of your simulation. Some common flowchart modules found in the "Basic Process" tab of the Project bar are the following elements: CREATE, PROCESS, DECIDE, DISPOSE, BATCH, SEPARATE, and ASSIGN and RECORD. To use these modules, you simply drag the flowchart module needed into the model and then you connect the modules together in the Model window flowchart view. For example, if I were to create our conceptual model of the IT service desk ticket queue, it would look like this (Figure 1).

As you see in our figure, we used the CREATE, PROCESS, and DISPOSE modules to illustrate the logic of the queue. Once a service desk ticket is created by the IT

Department (CREATE module), it is processed by the IT Department (PROCESS module), and it is closed by the IT Department (DISPOSE module). A bit confused? Rest assured, we have a whole chapter about this and it will get clearer as we take you step by step through an actual scenario.

For now, we are starting with a three-process scenario to get you thinking about simulation. This quick start model is provided on the companion site for download. For now, just think of it as creating a flowchart of your scenario. If you have used Microsoft Visio before, you will be right at home.

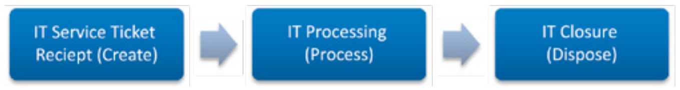


Figure 1. *IT service desk process*

Adding Data and Parameters to the Model

After creating the flowchart, the next step is to add data to each of the flowchart modules. You may assign the values for each module by double clicking on the modules in the model, which will open up a small dialog window. For example, for the CREATE module, let us say tickets arrive at an average of five per hour. You would enter that value directly into the CREATE module. Additionally, let us say tickets are processed and resolved at an average rate of 30 min. You would assign this value into your PROCESS module. We will provide you with a more detailed walk-through on how to do this later in this chapter.

Running the Simulation

After the model is complete, all you need to do is to select "Go" from the Run menu or press F5. There are other parameters that you may want to set up before running the simulation, such as the replication parameters where you can set the simulation period. But for the purpose of this quick introduction, we will just run the simulation.

Analyzing the Simulation

Arena provides a variety of reports so that you may analyze the simulation. You access the Reports panel from the Project bar.

CASE STUDY

There are a lot of interesting uses for simulations in security. One of them is evaluating the effect of security controls or mechanisms in your enterprise that otherwise would be difficult to recreate. For this chapter, let us put ourselves in the position of an Information Security Officer, who needs to evaluate different anti-virus (AV) e-mail security gateway offerings. One of the main things you will be concerned about is performance of the e-mail gateway device. Since the device will be sitting in-line and processing network traffic, you would want to make sure that the e-mail gateway is able to handle the volume of e-mails coming into your organization. Since this device will also sit in front of your e-mail server, there is no convenient way to test how the different e-mail security gateways will perform. This is where simulations come into play. Simulations give us a way to predict how a certain scenario or situation will play out based on available data. Of course, it will not be the same as testing the real thing, but it will at least provide us an estimate so that we can make an informed decision (Table1).

One of the first things we need for a simulation is data. Fortunately, in our scenario, a vendor (hereafter referred to as Vendor 1) provided us with a data set comparing its e-mail security gateway solution with products from other vendors (hereafter referred to as Vendor 2 and Vendor 3). You can download this data set from the book's website. Next, we will explain how this data set will be used in our scenario.

Table 1. *Vendor Scenario Data*

	Vendor 1	Vendor 2	Vendor 3
Average (s)	0.177271963	0.669560187	0.569069159
Test data (s)	0.0077	0.0119	0.5994
	0.0018	0.0201	0.5269
	0.0101	3.4405	0.4258
	0.0144	0.0701	0.5109
	0.0134	0.02	0.5619
	0.006	0.0119	0.5017
	0.1103	0.0012	0.4382
	0.0113	0.013	0.4346

0.0116	0.0161	0.4988
0.0185	0.0157	0.49
0.0021	0.2894	0.4843
0.0088	0.0089	0.4602
0.0051	0.0056	0.4431
0.0061	0.0067	1.4135
0.0106	0.0206	0.4199
0.0064	0.221	0.4332
0.01	0.025	0.4162
0.0128	3.067	0.4386
0.0113	1.098	0.4342
0.01	0.0158	0.4309
0.0058	1.145	0.4146
0.0023	0.112	0.4392
0.0126	0.0146	0.4678
0.0128	0.0098	0.4608
0.006	0.0201	0.4689
0.0064	0.0139	0.481
0.0088	0.0066	0.4449
0.011	1.945	0.4312
0.0142	0.8112	0.453
0.0058	0.855	1.2839
0.0062	0.874	0.445
0.0063	1.589	0.4275
0.014	0.0203	0.4517
0.0946	0.89	1.092
0.0011	0.0112	0.5119
0.0073	2.547	0.5966
0.0089	3.4003	1.2248
0.0111	2.3314	0.4345
0.0081	0.0158	0.5527
0.0114	0.0144	0.4991
0.0096	0.0204	0.4213
0.6305	0.0061	1.3264
0.0113	0.0105	0.4312
0.0059	2.578	0.4246
0.0102	0.95	0.4422
0.065	0.721	1.4509
0.0063	3.3614	0.478
0.0189	0.3078	0.4121
0.9503	3.3444	1.1532
0.0236	0.0103	0.4589
0.0094	0.00254	0.4124
0.0076	1.067	0.5074

0.0057	0.905	0.4509
1.0007	3.4747	0.4639
0.0061	0.0205	0.4729
0.0113	0.013	0.4343
0.0094	0.0018	0.4359
0.0061	0.0101	0.4761
0.0088	1.345	0.4594
0.0054	0.0936	0.6192
0.9407	3.7085	1.1916
12.2007	3.4655	0.5122
0.0035	1.523	0.4097
0.0028	0.0202	0.4422
0.0042	0.0147	0.4585
0.083	0.9678	1.282
0.0009	0.0059	1.4524
0.0078	0.0211	0.5503
0.0357	0.0496	0.7331
0.0068	0.016	0.4823
0.0107	0.0177	0.4378
0.0128	1.8678	0.4388
0.0113	0.013	0.4349
0.9457	0.812	0.9953
0.0109	0.0071	0.4457
0.0181	1.78	0.4099
0.0099	0.0102	0.4278
0.0066	0.832	0.4231
0.0111	0.0127	0.4346
0.0108	0.0144	0.4988
0.0159	0.0026	1.4738
0.0155	0.0772	0.4918
0.0113	0.0136	0.4157
0.0057	0.0101	0.4327
0.0064	0.0125	0.5496
0.0126	0.0146	0.4308
0.0171	0.042	0.4525
0.0038	0.1454	0.6053
0.0059	1.89	0.4243
0.0043	0.0407	0.7431
0.0066	0.8901	0.4764
0.0069	0.8542	0.4635
0.01	0.0059	0.522
0.0064	1.956	0.4802
0.0119	1.993	0.4333
0.0113	1.432	0.4343

0.0111	0.0127	0.4347
0.0064	0.0125	0.5521
0.065	0.711	1.2924
0.0912	0.0056	0.5121
0.0059	0.0107	0.4661
0.0125	0.0124	0.4177
0.0113	0.013	0.4157
0.8998	1.9081	0.4626
0.0059	0.0102	0.4304
0.0184	1.145	0.4216
0.0099	0.0144	0.5201

Table 2. *Vendor Processing Time – Overall Performance*

	Average Processing
Vendor 1	0.177271963
Vendor 2	0.669560187
Vendor 3	0.569069159

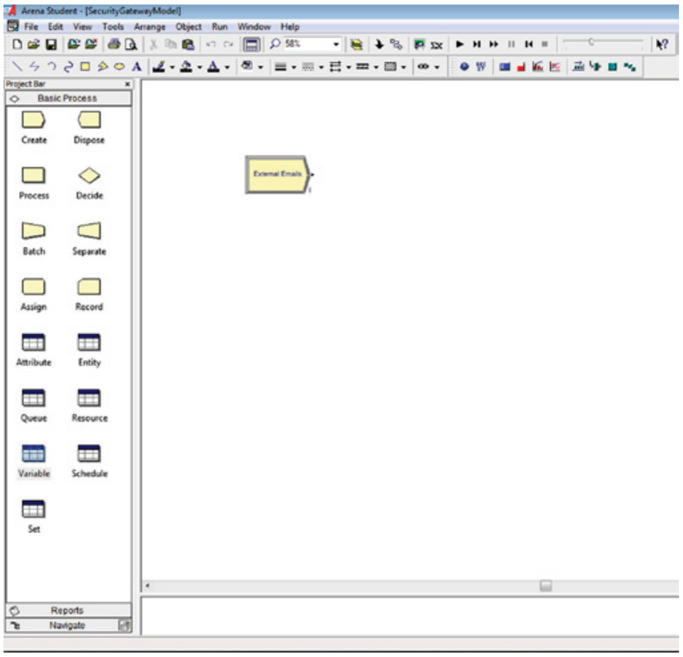


Figure 2. *Inserting the CREATE module*

Vendor 1 ran malicious e-mails through its e-mail security gateway and computed how fast the gateways processed the malicious e-mails (e.g., how fast the malicious e-mails were detected). Since Vendor 1 provided the data, as expected in terms of average processing times, Vendor 1 had an extremely short processing time (Table 2).

You may be asking yourself, how do we validate these numbers? Typically you would just take this data at face value and accept these numbers. However, what if you wanted to dive deeper to see if these are actually accurate for your organization's situation? This is where the fun part starts because we can do this through simulation. Let us dive into Arena!

First off, let us deconstruct our scenario. We need three components to start our simulation:

- First, we need to create the e-mails;
- Second, we need to create the 'e-mail security gateway' to process these e-mails; and
- Third, we need to create the inboxes that will receive the e-mails.

Fortunately, creating all of these components is fairly easy to do in Arena. Let us start first by creating a stream of e-mails that will come into our organization. This can be done by using the CREATE module (Figure 2).

One of the most important things that we need to do for a simulation is to create objects that will flow through the simulation that we are creating. In our scenario, the objects flowing through the system are the e-mails that will go through our security devices. In Arena, these objects are known as "entities." To be able to create entities, we need a CREATE module.

To make a CREATE module, all you have to do is drag the icon named create from the left-hand Basic Process bar to your work area. Your work area should look similar to Figure 3 below. It still looks a little sparse right now but this is only our first step.

Once you have added the CREATE module, the next step is to start configuring the attributes and properties for that module. To assign value to attributes or properties of the module, double click on the CREATE shape so that a dialog window appears, as shown in Figure 4.

In the dialog box, assign any name describing the entity being created. In this case, we labeled the entities as external e-mails. Let us change the entity type to "E-mail"

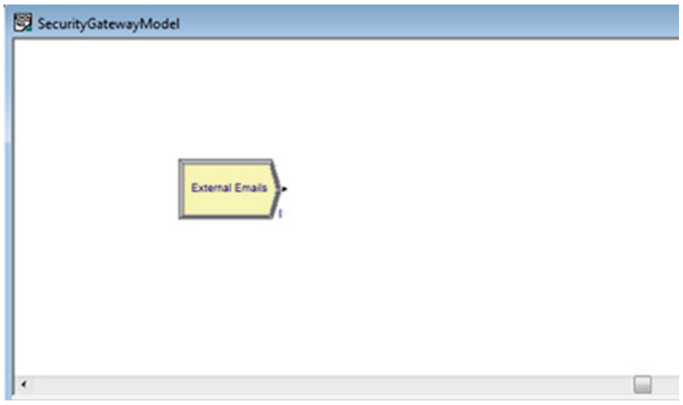


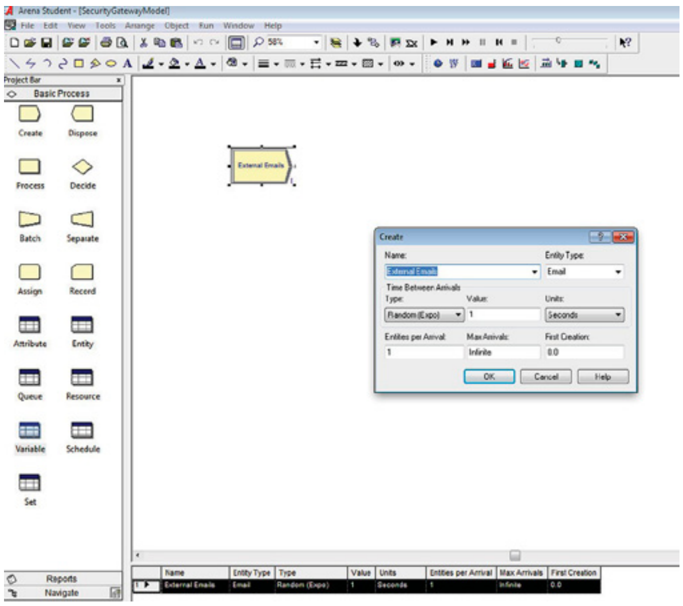
Figure 3. *Using CREATE to create external e-mail entities*

as well. We will also tell the simulation the average rate of e-mail arrival. The arrival of e-mails could be different for each organization. There are different ways of estimating this information (i.e., looking through your logs), but for the purposes of this example, we shall assume that on average an e-mail arrives every second. We can do this by changing the following:

- Type: Random (Expo)
- Value: 1
- Units: Seconds
- Entities per Arrival: 1
- Max Arrivals: Infinite
- First Creation: 0.0.

At this point, we have created entities for our simulation. This means that e-mails can now enter our system. But where will it go? Right now, nowhere. We need these e-mails to be processed, so we will need to create a process. This is done by dragging the PROCESS module from the left-hand navigation bar into the workplace as illustrated in Figure 5.

Since the e-mails going through the system need to be processed by the AV gateway, we will pattern our process to our gateway. In our simulation, the PROCESS module will represent the AV gateway that will be processing the external e-mails. Similar to what we did with the CREATE module, we will configure the attributes and properties of the PROCESS module. Open the dialog box for the PROCESS module in the same way you did with the CREATE module by double clicking on it.



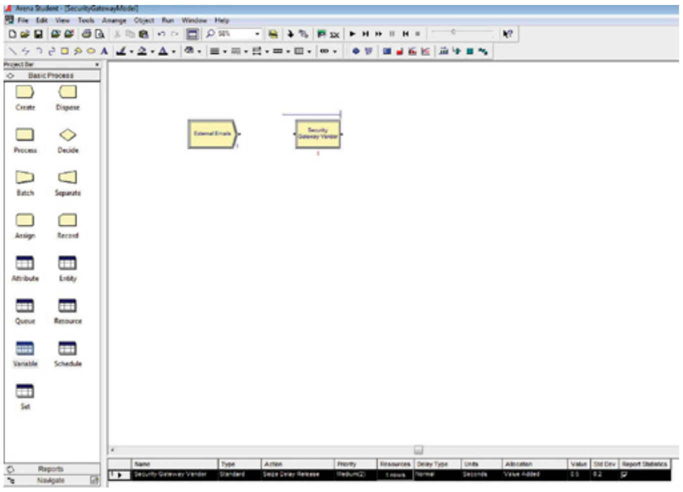
Create

Name: External Emails
Entity Type: Email
Time Between Arrivals
Type: Random (Expo)
Value: 1
Units: Seconds
Entities per Arrival: 1
Max Arrivals: Infinite
First Creation: 0.0

	Name	Entity Type	Type	Value	Units	Entities per Arrival	Max Arrivals	First Creation
1	External Emails	Email	Random (Expo)	1	Seconds	1	Infinite	0.0

Figure 4. *Updating the properties of the CREATE module*

First, let us assign a name for the process module. For this example, we will name it "Security Gateway Vendor."



	Name	Type	Action	Priority	Resources	Delay Type	Units	Allocation	Value	Std Dev	Report Statistics
1	Security Gateway Vendor	Standard	Seize Delay Release	Medium(2)	1 rows	Normal	Seconds	Value Added	0.5	0.2	✓

Figure 5. *Adding the PROCESS module*

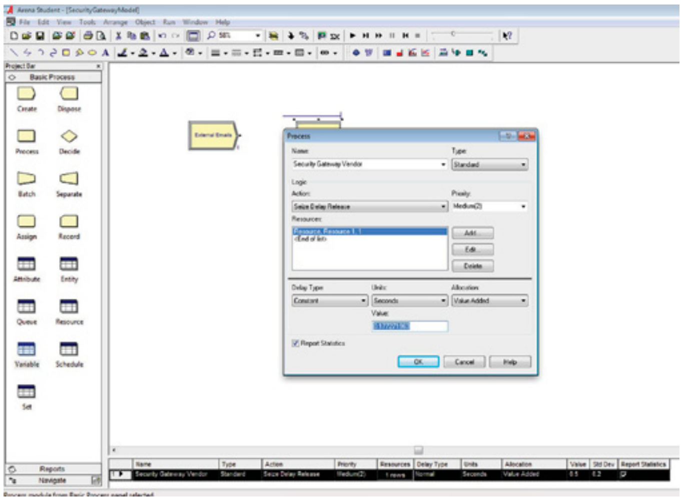
Next we will set the ACTION for this element. In the action drop down, we will select "Seize, Delay, Release" action. This means that when an e-mail arrives, it will wait until the resource becomes available and seize the resource, it will wait for the service interval, and then release the resource. This is essentially how an e-mail gateway operates: before a gateway sends an e-mail to the inbox, it will seize, delay (because of processing), and then release either to a user's inbox or a quarantine.

The "Delay" is an important value here in our simulation because it is actually the processing time. Relative to our scenario, this is the length of time the security gateway takes to process an e-mail to find out whether it is malicious or not.

Our next step is to customize our scenario. Since we have the vendor results on the average processing time, let us put the average processing value of Vendor 1 for this example. Your dialog box should look similar to Figure 6 and would have the following parameters:

- Name: Security Gateway Vendor
- Type: Standard
- Action: Seize Delay Release
- Priority Medium
- Resources: Resource, Resource 1,1
- Delay Type: Constant
- Units: Seconds
- Allocation: Value Added
- Report Statistics: Checked
- Value: 0.1777271863.

The next step is to create the resource for our security gateway. Since we are only going to be using one security



Process

Name: Security Gateway Vendor
Type: Standard

Logic
Action: Seize Delay Release
Priority: Medium(2)
Resources: Resource, Resource 1, 1
<End of list>

Delay Type: Constant
Units: Seconds
Allocation: Value Added
Value: [illegible]

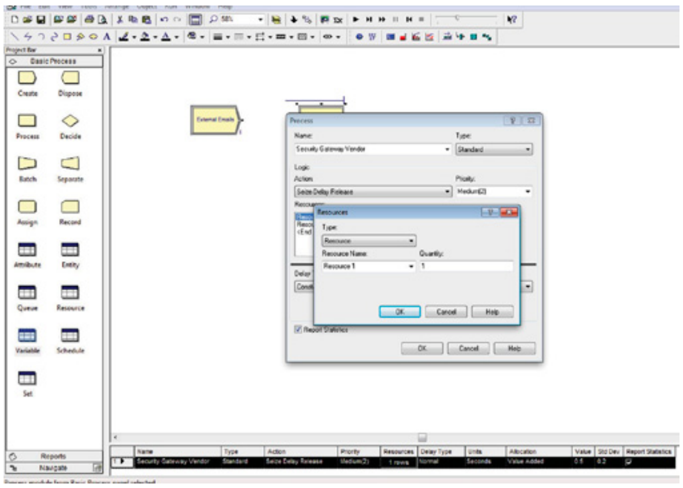
✓ Report Statistics

Figure 6. *Updating the properties of the PROCESS module*

gateway,v we will only create one resource. This setting is important if you are simulating multiple appliances or, in other cases, multiple processors. At this point, for simplicity, we will only create one resource, which can be done by clicking the "Add" button, which is located next to the Resource box. Your Resource dialog box should have the following parameters:

- Type: Resource
- Resource: 1
- Quantity: 1.

As the last step in our PROCESS module, we need to ensure that the CREATE module and the PROCESS module are linked together. In our scenario, this ensures that the e-mails created by the CREATE module goes to the PROCESS module to be processed by our AV gateway (Figure 7). Typically, Arena does this automatically;



Resources

Type: Resource
Resource Name: Resource 1
Quantity: 1

Figure 7. *Updating the resource property of the PROCESS module*

however, if it does not, click the Connect button in the upper toolbar of Arena to link both modules as seen in Figure 8.

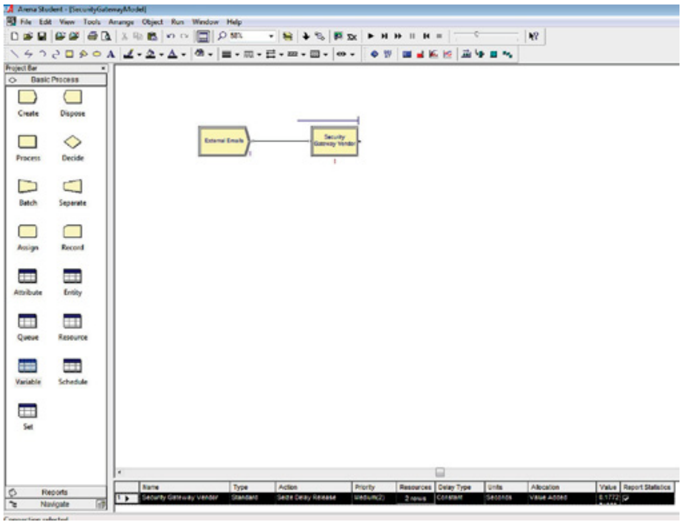


Figure 8. *Connect the CREATE module with the PROCESS module*

Finally, after processing, we need somewhere for the e-mails to go. This is where we use the DISPOSE module. Drag a DISPOSE module into your work area and label it as "Mailboxes." Then, connect the PROCESS module with the DISPOSE module. This means that after processing, the e-mails go to the mailboxes.

At this point, you are probably thinking that something is amiss with this scenario. Why would all processed e-mails go directly to the inboxes, right? You are absolutely right that something is amiss. For the sake of keeping our step-by-step tutorial simple, let us work with what we have for now. We will continue to expand on our scenario to make it more realistic. Your final simulation should look similar to the model in Figure 9.

Now that we have our simulation model, we are ready to run our first simulation. Before running our simulation, you will need to configure the different settings for the simulation. Since a simulation is technically trying to recreate a real-world scenario, we need to set up how long and how frequently we would like to let the scenario run.

This is fairly easy to do in Arena. Just click on Run (it is a selection on the top bar) and select Run Setup. For this simulation let us run it three times for 7 days, 24-hours a day. Since e-mails arrive at a one second interval, the base time unit will need to be changed to seconds. You will see a dialog box similar to Figure 10, in which you will add the following parameters:

- Number of Replications: 3
- Initialize Between Replications: Statistics and System Checked

- Warm-up Period: 0.0
- Replication Length: 7
- Hours Per Day: 24
- Base Time Units: Seconds
- Time Units: Hours
- Time Units: Days
- Termination Condition: Leave Blank.

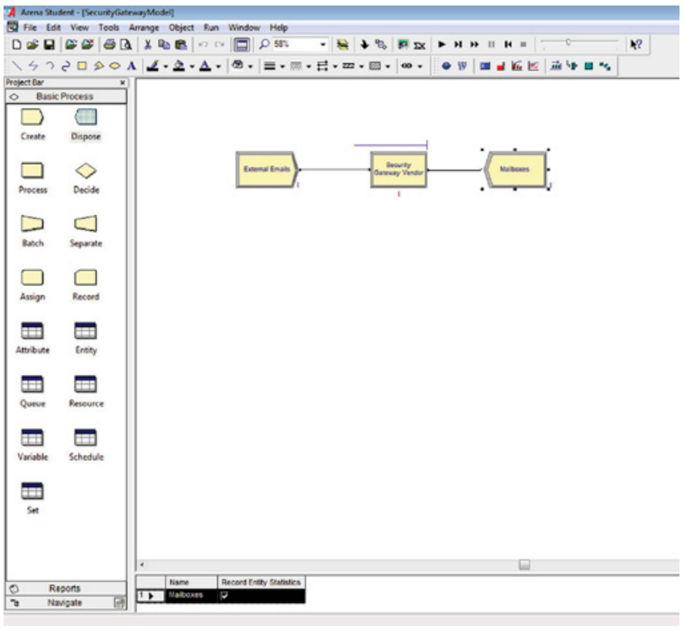


Figure 9. *Connect the CREATE module with the PROCESS module*

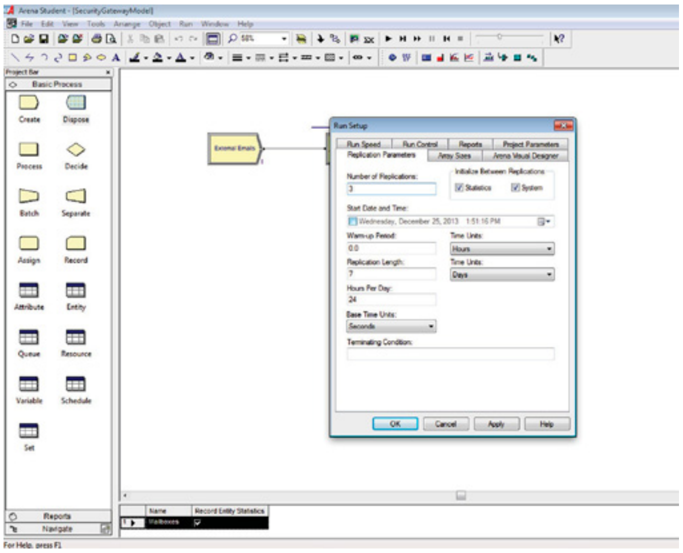
After the Run parameters have been configured, we will add some information on the Project Parameters to describe the project by clicking on the Project Parameter tab (see below figure). We are now ready to run our simulation. You do this by simply clicking Run, then selecting Go.

- Project Title: Security Gateway
- Project Description: Add any description

After clicking Go, the simulation will animate and there will be elements moving. You will see "e-mails" coming from the CREATE module (external e-mails), moving to the PROCESS module (security gateway), and being accepted by the DISPOSE module (inbox) (Figure 11).

Congratulations, you have now completed your first simulation! The simulation may take some time to process before we get the results. Unfortunately, even with setting at the highest speed, running three simulations on e-mails over a 7-day period will take some time to process.

Fortunately, Arena has a feature, which allows what is called "batch processing." Batch processing bypasses all of the animation, which speeds up processing. To do this, you first need to stop the simulation by clicking on Run,


Run Setup

Number of Replications: 3

Initialize Between Replications: ✓ Statistics ✓ System

Start Date and Time: Wednesday, December 25, 2013 1:51:16 PM

Warm-up Period: 0.0

Time Units: Hours

Replication Length: 7

Time Units: Days

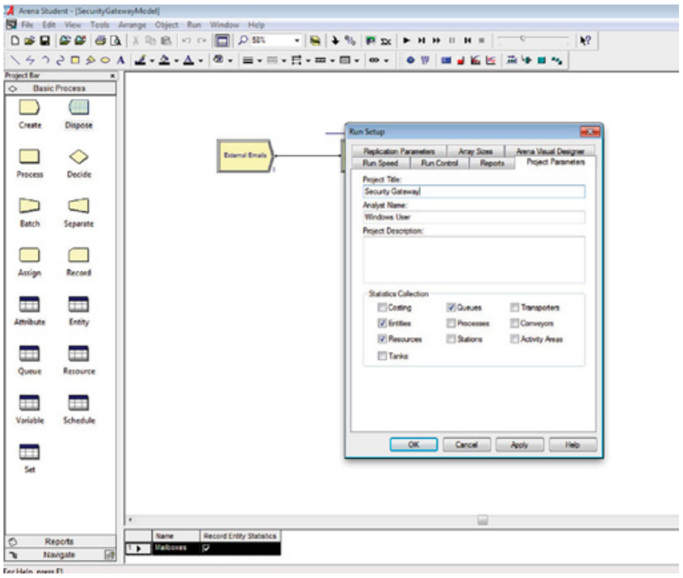
Hours Per Day: 24

Base Time Units: Seconds

Terminating Condition:

Figure 10. *Run parameters set up*

and then End. Next, select Run Control and click on Batch Run (No Animation). By doing this, you will speed up the simulation so that you can generate the results faster (Figure 12).


Run Setup

Project Title: Security Gateway

Analyst Name: Windows User

Project Description:

Statistics Collection: Costing ✓ Queues Transporters ✓ Entities Processes Conveyors ✓ Resources Stations Activity Areas Tanks

Figure 11. *Project set up*

Let us try running the simulation again using these new parameters. You will notice this time that you see no animation and you immediately receive the results.

Your output will be a report, including some interesting values such as minimum averages, maximum averages, minimum values, and maximum values. Basically, these values are the descriptive statistics for your simulation processing times, which we ran three times in 7-day increments (Figure 13).

The real value of the simulation is seen once we start comparing the vendors. Let us try doing this next. For each vendor, change the delay value to match each vendor's average processing time. If you gather the results, you should obtain the results in Table 3. Your results should show that Vendor 1's claims are accurate: on average, Vendor 1 shows the best performance.

For now, we will accept that Vendor 1's claim is correct. However, as you know, statistics can sometimes be interpreted to provide misleading information. Let us start by going back and look at the original data that the vendor gave us.

A very interesting point about the original data is that the vendor provided the actual results of their testing. With the results for individual processing times for each of the e-mails, instead of just the average processing time for all of the e-mails, we can use a very simple yet relatively well-known technique called standard deviation (SD).

SD shows the variation or dispersion that exists in relation to the mean (also called average). A low SD indicates the data points tend to be close to the mean (also called expected value); a high SD indicates the data points are spread out over a larger range of values.

Let us start with some simple spreadsheet work. We want to open the file containing the sample data and obtain the SD of the data by using the STDEVP function (=STDEVP). This explanation may be somewhat

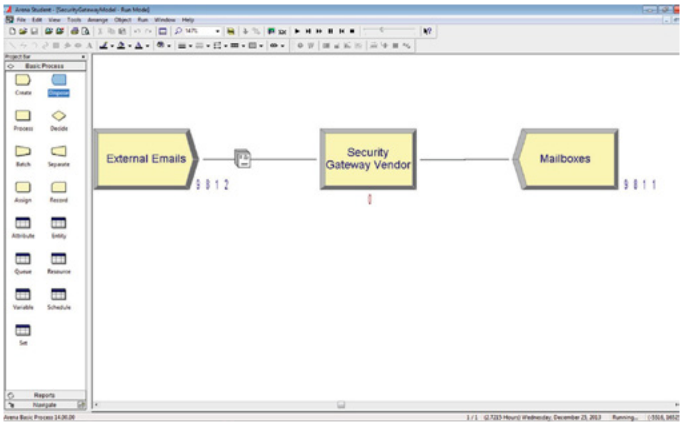


Figure 12. *A running simulation*

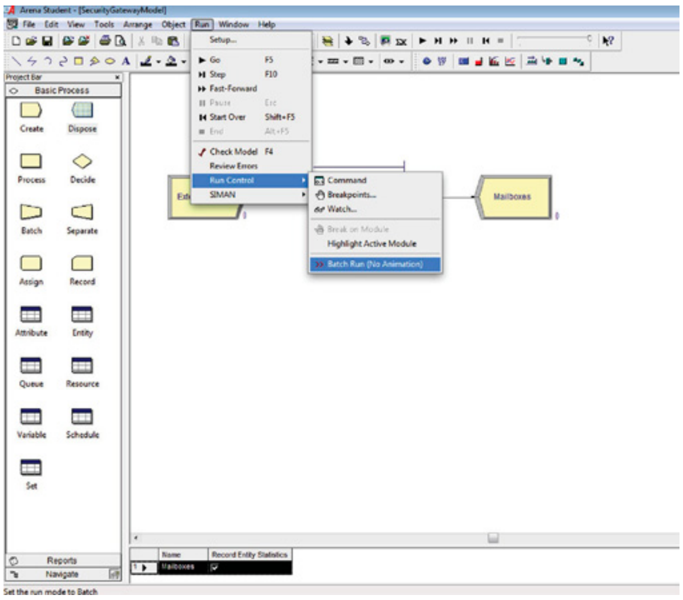
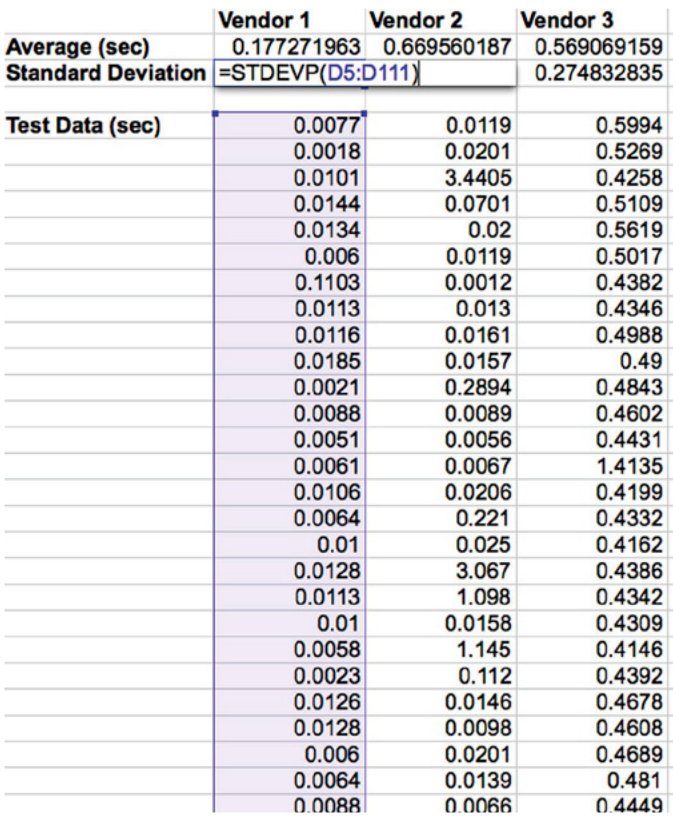


Figure 13. *Doing a batch run*



	Vendor 1	Vendor 2	Vendor 3
Average (sec)	0.177271963	0.669560187	0.569069159
Standard Deviation	=STDEVP(D5:D111)		0.274832835
Test Data (sec)	0.0077	0.0119	0.5994
	0.0018	0.0201	0.5269
	0.0101	3.4405	0.4258
	0.0144	0.0701	0.5109
	0.0134	0.02	0.5619
	0.006	0.0119	0.5017
	0.1103	0.0012	0.4382
	0.0113	0.013	0.4346
	0.0116	0.0161	0.4988
	0.0185	0.0157	0.49
	0.0021	0.2894	0.4843
	0.0088	0.0089	0.4602
	0.0051	0.0056	0.4431
	0.0061	0.0067	1.4135
	0.0106	0.0206	0.4199
	0.0064	0.221	0.4332
	0.01	0.025	0.4162
	0.0128	3.067	0.4386
	0.0113	1.098	0.4342
	0.01	0.0158	0.4309
	0.0058	1.145	0.4146
	0.0023	0.112	0.4392
	0.0126	0.0146	0.4678
	0.0128	0.0098	0.4608
	0.006	0.0201	0.4689
	0.0064	0.0139	0.481
	0.0088	0.0066	0.4449

Figure 14. *Computing SD in a spreadsheet*

Table 3. *Vendor Processing Time – Initial Simulation Run*

	Processing Time	Average	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Vendor 1	0.177271963	0.01911149	0.01897183	0.01919783	0.00	0.9207
Vendor 2	0.669560187	0.6809	0.6760	0.6863	0.00	11.6437
Vendor 3	0.569069159	0.3770	0.3740	0.3740	0.00	8.3927

confusing, so see Figure 14. Open the next tab of the sample data containing the computation.

Table 4. *Vendor Processing Time – Adding SD*

	Average Processing		Standard Deviation
Vendor 1	0.177271963		1.185744915
Vendor 2	0.669560187		1.026043254
Vendor 3	0.569069159		0.274832835
	Vendor 1	Vendor 3	Vendor 3
Average (s)	0.177271963	0.669560187	0.569069159
Standard deviation	1.185744915	1.026043254	0.274832835

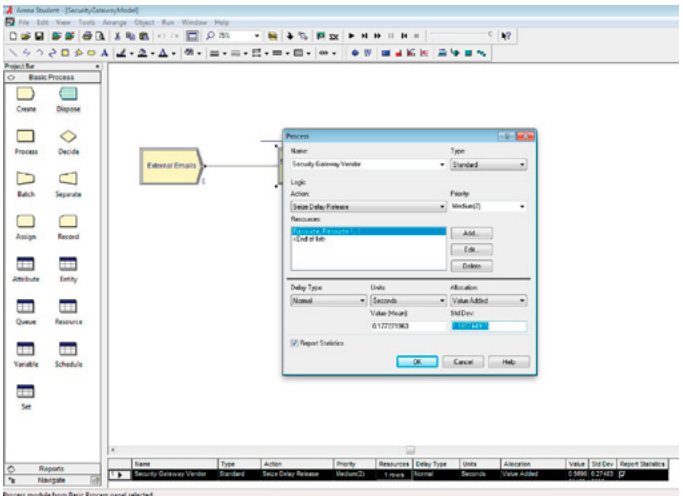
After computing the SD for all of the vendors, we see that Vendor 1 actually has a big SD. This means that the results of the test data vary greatly. For example, it processes e-mails very fast in some cases, but in other cases, it processes e-mails very slowly. You may be asking yourself, what exactly does this tell us? Obviously, those of you who understand SDs probably already have an inkling of what this means, but we will run a simulation so we can see what our scenario generates (Table 4).

Now, we will go back to the simulation to enter our newly computed values. Click on the PROCESS module. However, let us change things up a bit. Instead of using the "Constant" delay type, we will use the "Normal" delay type or what we call a normal distribution. The normal distribution is a function telling you the probability that an observation, in some context, will fall between any two real numbers.

In the PROCESS dialog box, we will maintain the mean value, but we will now add the SD for the vendor. The entries you select should be similar to the below values, which will be put in the dialog box (See Figure 15). Next, we will run the simulation.

- Name: Security Gateway Vendor
- Type: Standard
- Action: Seize Delay Release
- Priority: Medium
- Resource: Resource, Resource 1,1
- Delay Type: Normal
- Units: Seconds
- Allocation: Value Added
- Value: 0.177271963
- Std Dev: 1.185744915
- Report Statistics: Checked.

We will run the simulation for all the vendors and collect their results. Remember to make the change to "Normal" for all of the vendors and to add in the SD. Once you have run everything and collected the results, your results should be similar to values in the tables below.


Process

Name: Security Gateway Vendor
Type: Standard
Action: Seize Delay Release
Priority: Medium(2)
Resources: Resource, Resource 1, 1
Delay Type: Normal
Units: Seconds
Allocation: Value Added
Value (Mean): 0.177271963
Std Dev: [illegible]
[✓] Report Statistics

Figure 15. *Updating the PROCESS Dialog's standard deviation*

You should now notice that the results are quite interesting. The values have changed a considerable amount because we used the normal distribution. In fact, Vendor 1 did not perform as well as expected with these results. In this scenario, Vendor 3 actually had better results.

The reason for this is that Vendor 3 had more consistent results. The processing times for Vendor 3 were more stable and, more importantly, less variable. Conversely, Vendor 1 had a lot of variability, which greatly affected the overall processing times. This is why it is important to understand what you are processing and how it will affect your results. Had you just gone with the vendor results, you would not have known this information – this provides you with value-added information, which could affect your choice of a vendor (Table 5).

For the final part of our tutorial, we will extend our simulation model to make it more detailed and realistic. In the previous scenario, we assumed that all e-mails were malicious, but in reality we would never do this. For a more realistic scenario, we will incorporate the DECIDE module to create conditional branches.

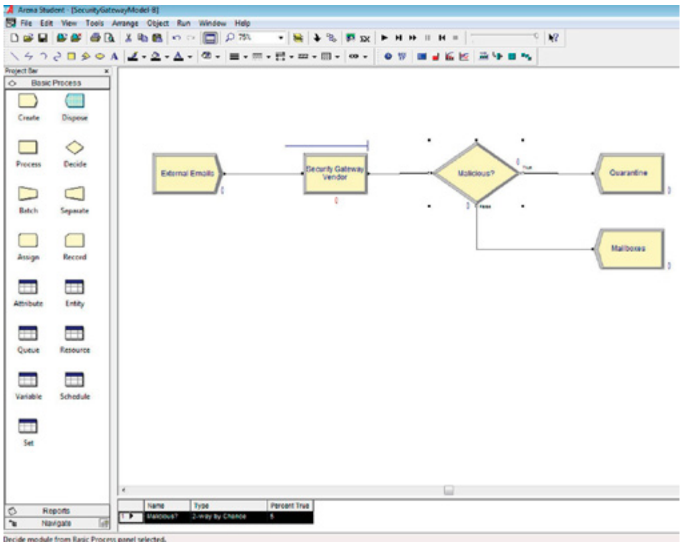
The DECIDE module can be found in the Basic Process tab, which is located on the left-hand side of our work area. The DECIDE module helps us to create conditions (also known as "if-then" conditions) that are similar to what you would see in a flowchart (Figure 16).

We will now create a scenario with conditional elements. As we already mentioned, not all e-mails will have malicious attachments. Let us say only 5% of all e-mails will have malicious attachments. How did we get 5%? This is entirely dependent on you, but to have

Table 5. Vendor Processing Time – Additional SD Settings

	Average	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Vendor 1	1.0290	1.0231	1.0342	0.00	25.1238
Vendor 2	3.8393	3.8183	3.8531	0.00	46.5414
Vendor 3	0.4661	0.4650	0.4680	0.00	9.4981
Vendor 1	0.01911149	0.01897183	0.01919783	0.00	0.9207
Vendor 2	0.6809	0.6760	0.6863	0.00	11.6437
Vendor 3	0.3770	0.3740	0.3740	0.00	8.3927

a more realistic scenario, you should probably try to check industry benchmarks. For example, Symantec issues a monthly intelligence report similar to the one in this link where you can find benchmarks: *http://www.symantec.com/content/en/us/enterprise/other_resources/b-intelligence_report_07-2014.en-us.pdf*.



	Name	Type	Percent True
1	Malicious?	2-way by Chance	5

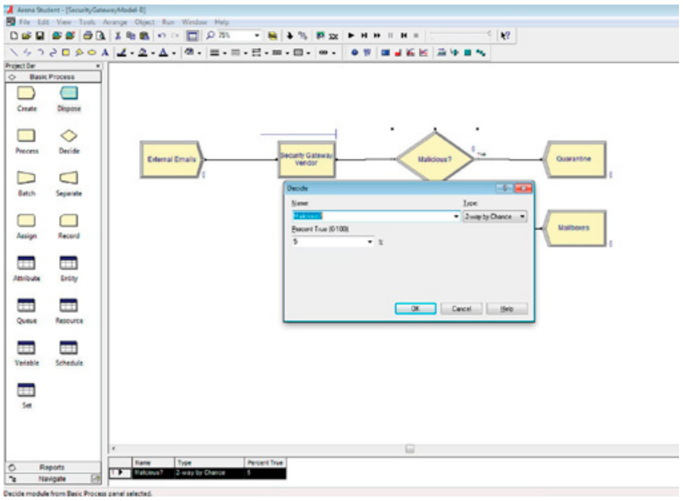
Figure 16. *Adding a DECIDE module in the simulation*

Now, let us go back to our workspace. Drag a DECIDE module into the work area, and double click on the module. Once you are at the dialog box, type in a name and change the Type to "2-way chance," which is the default. Since there's a 5% chance of a mail being malicious, you will enter 5% in the Percent True text box. Your entries should be similar to the parameters shown in Figure 17.

Finally, we will close of the system by adding a DISPOSE module for both the True and False branches. Note that all simulations must have a DISPOSE module. Let us label the DISPOSE modules as Quarantine for True then Mailbox for false. This will be a little understandable when we start talking about counters.

- Name: Malicious?
- Type: 2-way by Chance
- Percent True (0–100): 5%.

You should be familiar with the RECORD module, which acts like an advance counter. This module is used to run different computations and to store the processed results within the module. For this scenario, let us make a simple counter using the RECORD module to track clean and malicious e-mails. If an e-mail is malicious, then we will assume the action to be taken is quarantining the e-mail. If the e-mail is clean, the action to be taken is to send it to the user's inbox. Therefore, we will make two counters: one is a Quarantine counter and the other is a Mailbox counter.



Name: Malicious?
Type: 2-way by Chance
Percent True (0-100): 5 %

	Name	Type	Percent True
1	Malicious?	2-way by Chance	5

Figure 17. *Updating the properties of a DECIDE module*

We do this by connecting the RECORD modules into the DECIDE module, similar to Figure 18. As with all simulations, all paths should have an end point. So, you need to remember to create DISPOSE modules for the two paths: one for the clean e-mails and one for the malicious e-mails.

We will now configure the parameters of our RECORD module. Double click on each of the RECORD modules and set it to "Count" with a value of 1. This means that if the e-mail was malicious, then the RECORD module for the Quarantine RECORD module will be increased by 1. You would do the same for the Mailbox RECORD module. In the context of this scenario, the following applies: if the e-mail is malicious (therefore, YES), then the counter for malicious e-mails will be incremented by 1. If the e-mail is not malicious, then the clean e-mail counter will be incremented by 1 (Figure 19).

- Name: Malicious E-mail
- Type: Count
- Value: 1
- Record into Set: Unchecked
- Counter Name: Malicious E-mail.

That is it – our simulation is now complete! We have created a more complex simulation, which utilized the DECISION and RECORD modules. All you have to do now is to run the simulation and wait for the reports to be generated (Figure 20).

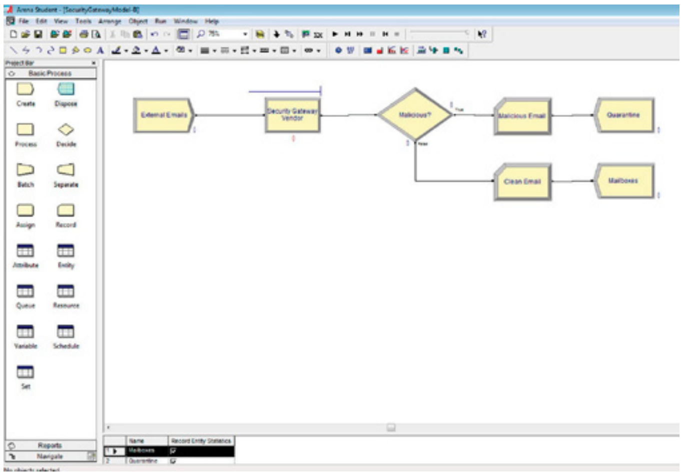
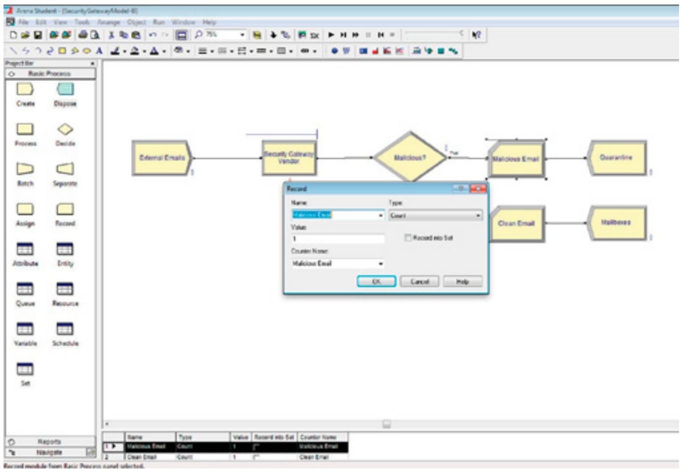


Figure 18. *Creating RECORD and DISPOSE modules*

As you can see, our simulation is slowly getting more advanced. However, we are still not finished with it. What about efficacy? The vendor actually provided us with efficacy information and we can use this information to improve our simulation. The question is how do we incorporate this information into our simulation (Figure 21)?

In our previous simulation, we assumed that all e-mails that were considered clean were actually clean but in reality, things seldom work this way because malicious e-mail will get through AV checks. This is why the vendor provided us with the ratings of efficacy for each of the products being



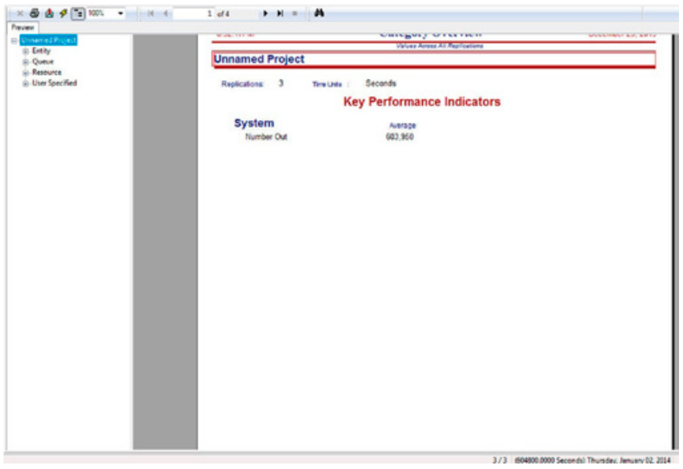
Record

Name: Malicious Email
Type: Count
Value: 1
Record into Set
Counter Name: Malicious Email

Figure 19. *Updating the properties of the RECORD module*

reviewed. Next, we need to add another conditional element in the simulation so that we may include this process.

We will add another DECIDE module to the second filter for the clean decision, but this time, we will add a condi-



Values Across All Replications

Unnamed Project

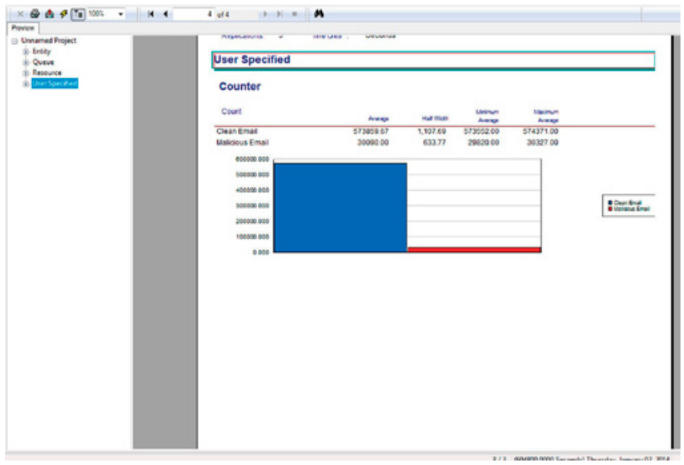
Replications: 3 Time Units: Seconds

Key Performance Indicators

System Average
Number Out 603,960

3 / 3 (604800.0000 Seconds) Thursday, January 02, 2014

Figure 20. *Viewing the report*



User Specified

Counter

Count

	Average	Half Width	Minimum Average	Maximum Average
Clean Email	573859.67	1,107.69	573552.00	574371.00
Malicious Email	30090.00	633.77	29820.00	30327.00

Figure 21. *Additional report information*

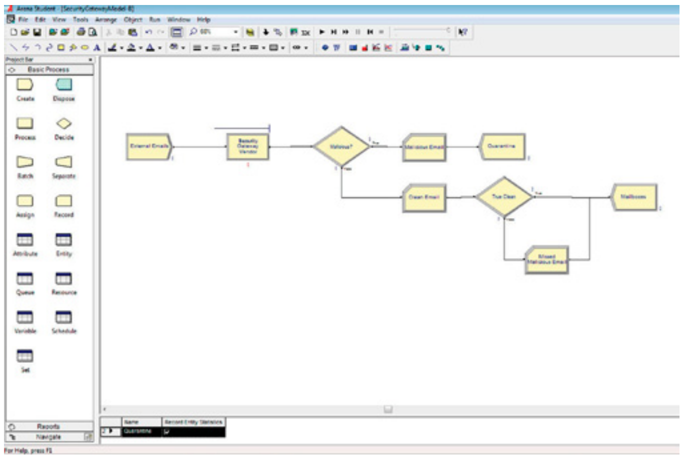


Figure 22. *Removing false negatives through another DECISION module*

tion for every clean e-mail. In this updated scenario, once the decision is made that an e-mail is clean, we place another decision regarding "how sure are we that the e-mail is clean." We will call this our "True Clean" decision box, which is a layer to show the probability that a clean e-mail is actually clean. By adding this decision box, we are able to provide a means to determine "false negatives" or malicious e-mails that were missed by the security gateway. Your updated simulation should look similar to Figure 22.

Table 6. *Vendor Processing Time – Including Efficacy*

Vendor 1	99.90%
Vendor 2	99.70%
Vendor 3	98.40%

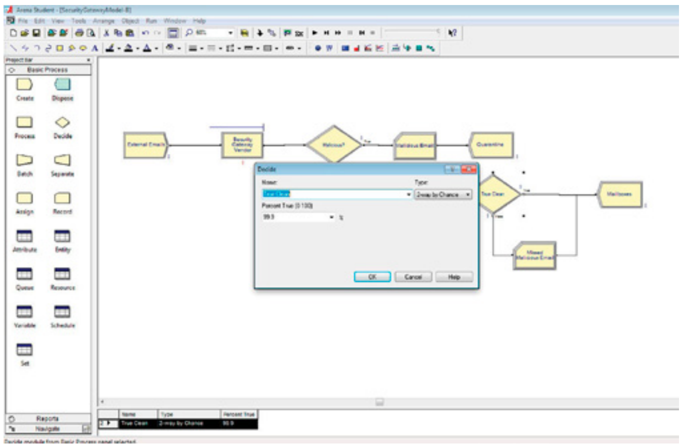
We will now configure our new DECISION module. Double click the True Clean box and add the efficacy rating into the Percent True box. This will simulate the probability that the e-mail is actually clean. We then add a counter to "how many e-mails that were considered clean were actually malicious." We will use the RECORD module to add a "Missed Malicious E-mail" box. Below are the e-mails that the AV missed, where the vendor's verdict was clean but the e-mails were actually malicious (Table 6).

Using our vendor spreadsheet, if the vendor's security gateway has a 99.9% efficacy then we put 99.9% in the Percent True. These values will allow us to compute the probability of an e-mail actually being clean which equates to the efficacy in our simulation. See Figure 23.

- Name: True Clean
- Type: 2-way by Chance
- Percent True (0–100): 99%.

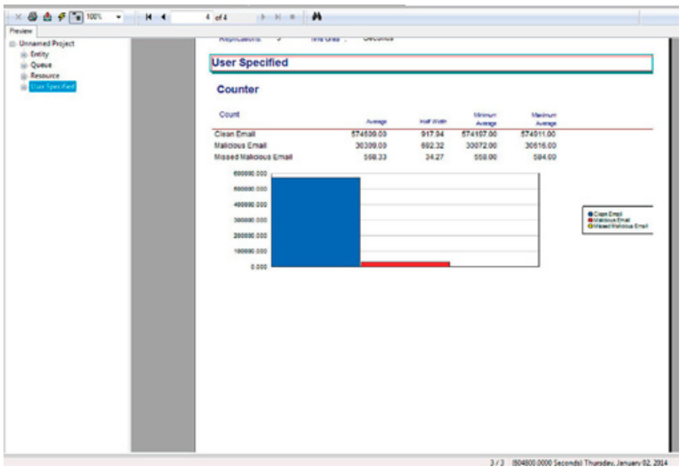
Finally, let us run our simulation and wait for our report! We will do this for all of our vendors. Remember to make changes to the average processing times, the SD and the efficacy of each simulation.

As we wrap up this chapter, let us go through the completed simulation statistics in Figure 24. In summary, here are the observations we obtained from our simulation.



Name: True Clean
Type: 2-way by Chance
Percent True (0-100): 99.9

Figure 23. *Adding efficacy into the simulation*



User Specified

Counter

```
Count
                                    Average    Half Width    Minimum      Maximum
                                                             Average      Average
Clean Email                       574509.00      917.94    574107.00    574911.00
Malicious Email                    30309.00      692.32     30072.00     30616.00
Missed Malicious Email               568.33       34.27       558.00       584.00
```

Figure 24. *Viewing the final report*

- Vendor 3 is actually pretty good in terms of performance. When we start looking at the efficacy (i.e., 99.9% vs 98%) and considering the amount of e-mails processed in a week, the difference between a 99.9% efficacy and a 98% efficacy rate is a staggering amount. The difference can be as large as 8000 malicious e-mails!
- Even a 99.9% efficacy would result in 568 malicious e-mails, which is still a lot of malicious e-mails. This shows that even when a vendor's AV is used, there is still a big chance that one of your employees could be infected.

	Average	Minimum Average	Maximum Average	Minimum Value	Maximum Value	Average Processed in a week	Average Missed Malicious E-mail
Vendor 1	1.0290	1.0231	1.0342	0.00	25.1238	604,918	568.33
Vendor 2	3.8393	3.8183	3.8531	0.00	46.5414	605,514	1704.00
Vendor 3	0.4661	0.4650	0.4680	0.00	9.4981	605,311	9284.00

The following tables provide a summary of the results we collected during the simulation:

Average Processing Times

	Average Processing
Vendor 1	0.177271963
Vendor 2	0.669560187
Vendor 3	0.569069159

In this chapter, we demonstrated how it is possible to simulate performance when it is difficult to test a system or otherwise obtain results. In our security scenario, we simulated an AV gateway for three vendors; however, there are a lot of other interesting uses for simulations. Another possible use of simulations in security could be recreating virus propagation within a network to see how fast it will affect your enterprise. You could also use simulations to see the effects of patching, of re-imaging machines and of AV updates. On a larger scale, simulations could be used to demonstrate cyber attacks against your organization. You can create a simulation representing your whole network, including firewalls, a intrusion prevention system and network segments, to see how attacks would or would not be detected, among other things. In conclusion, simulations in security are particularly useful in evaluating the effect of security controls or mechanisms in your enterprise that would, otherwise, be difficult to recreate.

Using a Constant Delay Type

	Average	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Vendor 1	0.01911149	0.01897183	0.01919783	0	0.9207
Vendor 2	0.6809	0.676	0.6863	0	11.6437
Vendor 3	0.377	0.374	0.374	0	8.3927

Using a Normal Distribution (STD)

	Average Processing	Standard Deviation
Vendor 1	0.17727196	1.18574492
Vendor 2	0.66956019	1.02604325
Vendor 3	0.56906916	0.27483284

	Average	Minimum Average	Maximum Average	Minimum Value	Maximum Value
Vendor 1	1.029	1.0231	1.0342	0	25.1238
Vendor 2	3.8393	3.8183	3.8531	0	46.5414
Vendor 3	0.4661	0.465	0.468	0	9.4981

Final Results

	Average	Minimum Average	Maximum Average	Minimum Value	Maximum Value	Avg. Missed Malicious E-mail
Vendor 1	1.029	1.0231	1.0342	0	25.1238	568.33
Vendor 2	3.8393	3.8183	3.8531	0	46.5414	1704
Vendor 3	0.4661	0.465	0.468	0	9.4981	9284

Data Used in the Simulation

	Vendor 1	Vendor 2	Vendor 3
Average (s)	0.177271963	0.669560187	0.569069159
Standard deviation	1.185744915	1.026043254	0.274832835
Test data (s)	0.0077	0.0119	0.5994
	0.0018	0.0201	0.5269
	0.0101	3.4405	0.4258
	0.0144	0.0701	0.5109
	0.0134	0.02	0.5619
	0.006	0.0119	0.5017
	0.1103	0.0012	0.4382
	0.0113	0.013	0.4346
	0.0116	0.0161	0.4988
	0.0185	0.0157	0.49
	0.0021	0.2894	0.4843
	0.0088	0.0089	0.4602
	0.0051	0.0056	0.4431
	0.0061	0.0067	1.4135
	0.0106	0.0206	0.4199
	0.0064	0.221	0.4332
	0.01	0.025	0.4162
	0.0128	3.067	0.4386
	0.0113	1.098	0.4342
	0.01	0.0158	0.4309
	0.0058	1.145	0.4146
	0.0023	0.112	0.4392
	0.0126	0.0146	0.4678
	0.0128	0.0098	0.4608
	0.006	0.0201	0.4689
	0.0064	0.0139	0.481
	0.0088	0.0066	0.4449
	0.011	1.945	0.4312
	0.0142	0.8112	0.453
	0.0058	0.855	1.2839
	0.0062	0.874	0.445
	0.0063	1.589	0.4275
	0.014	0.0203	0.4517
	0.0946	0.89	1.092
	0.0011	0.0112	0.5119
	0.0073	2.547	0.5966
	0.0089	3.4003	1.2248
	0.0111	2.3314	0.4345
	0.0081	0.0158	0.5527
	0.0114	0.0144	0.4991
	0.0096	0.0204	0.4213
	0.6305	0.0061	1.3264
	0.0113	0.0105	0.4312
	0.0059	2.578	0.4246
	0.0102	0.95	0.4422
	0.065	0.721	1.4509
	0.0063	3.3614	0.478
	0.0189	0.3078	0.4121
	0.9503	3.3444	1.1532
	0.0236	0.0103	0.4589
	0.0094	0.00254	0.4124
	0.0076	1.067	0.5074
	0.0057	0.905	0.4509
	1.0007	3.4747	0.4639
	0.0061	0.0205	0.4729
	0.0113	0.013	0.4343
	0.0094	0.0018	0.4359
	0.0061	0.0101	0.4761
	0.0088	1.345	0.4594
	0.0054	0.0936	0.6192
	0.9407	3.7085	1.1916
	12.2007	3.4655	0.5122
	0.0035	1.523	0.4097
	0.0028	0.0202	0.4422
	0.0042	0.0147	0.4585
	0.083	0.9678	1.282
	0.0009	0.0059	1.4524
	0.0078	0.0211	0.5503
	0.0357	0.0496	0.7331
	0.0068	0.016	0.4823
	0.0107	0.0177	0.4378
	0.0128	1.8678	0.4388
	0.0113	0.013	0.4349
	0.9457	0.812	0.9953
	0.0109	0.0071	0.4457
	0.0181	1.78	0.4099
	0.0099	0.0102	0.4278
	0.0066	0.832	0.4231
	0.0111	0.0127	0.4346
	0.0108	0.0144	0.4988
	0.0159	0.0026	1.4738
	0.0155	0.0772	0.4918

0.0113	0.0136	0.4157
0.0057	0.0101	0.4327
0.0064	0.0125	0.5496
0.0126	0.0146	0.4308
0.0171	0.042	0.4525
0.0038	0.1454	0.6053
0.0059	1.89	0.4243
0.0043	0.0407	0.7431
0.0066	0.8901	0.4764
0.0069	0.8542	0.4635
0.01	0.0059	0.522
0.0064	1.956	0.4802
0.0119	1.993	0.4333
0.0113	1.432	0.4343
0.0111	0.0127	0.4347
0.0064	0.0125	0.5521
0.065	0.711	1.2924
0.0912	0.0056	0.5121
0.0059	0.0107	0.4661
0.0125	0.0124	0.4177
0.0113	0.013	0.4157
0.8998	1.9081	0.4626
0.0059	0.0102	0.4304
0.0184	1.145	0.4216
0.0099	0.0144	0.5201

ABOUT THE AUTHOR

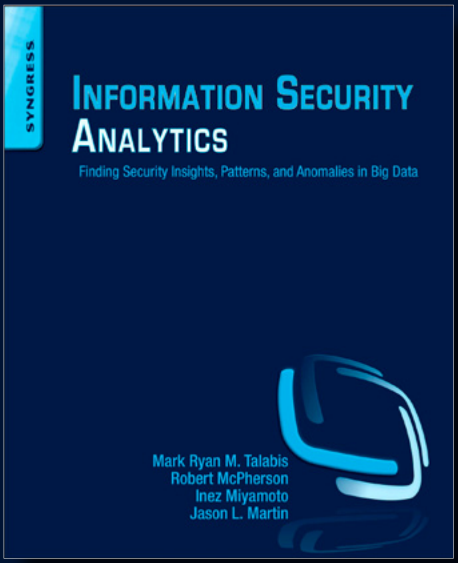
Robert McPherson leads a team of data scientists for a Fortune 100 Insurance and Financial Service company in the United States. He has 14 years of experience as a leader of research and analytics teams, specializing in predictive modeling, simulations, econometric analysis, and applied statistics. Robert works with a team of researchers who utilize simulation and big data methods to model the impact of catastrophes on millions of insurance policies…simulating up to 100,000 years of hurricanes, earthquakes, and wildfires, as well as severe winter and summer storms, on more than 2 trillion dollars worth of insured property value. He has used predictive modeling and advanced statistical methods to develop automated outlier detection methods, build automated underwriting models, perform product and customer segmentation analysis, and design competitor war game simulations. Robert has a master's degree in Information Management from the Harvard University Extension.

ABOUT THE AUTHOR

I. Miyamoto is a computer investigator in a government agency with over 16 years of computer investigative and forensics experience, and 12 years of intelligence analysis experience. I. Miyamoto is in the process of completing a PhD in Systems Engineering and possesses the following degrees: BS in Software Engineering, MA in National Security and Strategic Studies, MS in Strategic Intelligence, and EdD in Education.

ABOUT THE AUTHOR

Jason L. Martin is Vice President of Cloud Business for FireEye Inc., the global leader in advanced threat-detection technology. Prior to joining FireEye, Jason was the President and CEO of Secure DNA (acquired by FireEye), a company that provided innovative security products and solutions to companies throughout Asia-Pacific and the U.S. Mainland. Customers included Fortune 1000 companies, global government agencies, state and local governments, and private organizations of all sizes. He has over 15 years of experience in Information Security, is a published author and speaker, and is the cofounder of the Shakacon Security Conference.


SYNGRESS
INFORMATION SECURITY ANALYTICS
Finding Security Insights, Patterns, and Anomalies in Big Data
Mark Ryan M. Talabis
Robert McPherson
Inez Miyamoto
Jason L. Martin

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

Using FreeBSD as a File Server with ZFS

Ivan Voras

The ZFS storage workshop will teach you how to create a ZFS file system from scratch and build a file server on top of it, but it will also teach you how ZFS, file systems and storage servers work in general. You will learn what ZFS looks like, its many features and quirks, and how to use it in a FreeBSD server as a building block of a small file server.

ZFS is the ground-breaking file system originally developed at Sun Inc. for their Solaris operating system. It was open-sourced as a part of their OpenSolaris initiative and from there has spread to multiple other operating systems. FreeBSD was the first one to implement a working port, and though it has taken a fairly long time of tweaking and stabilization, it is now a robust and popular choice. There are products which successfully build upon the technologies of FreeBSD and ZFS, such as FreeNAS and its related enterprise-class products from iXsystems, which automate and simplify a lot of the tasks, but all of them use the same ZFS interface under the hood, which is not that complicated in itself.

The requirements for this workshop are decent knowledge of FreeBSD, a basic familiarity with command-line operations, and a system (possibly a virtual machine) on which the student will perform the required tasks, containing at least four hard drives (physical or virtual). Since the topic of this workshop is file servers, the participants must prepare a virtual or a physical machine with at least two disk drives (and preferably 4), which which to perform the exercises and the setup from the workshop.

http://bsdmag.org/course/using-freebsd-as-a-file-server-with-zfs-2/

Ivan Voras is a FreeBSD developer and a long-time user, starting with FreeBSD 4.3 and throughout all the versions since. In real life he is a researcher, system administrator and a developer, as opportunity presents itself, with a wide range of experience from hardware hacking to cloud computing. He is currently employed at the University of Zagreb Faculty of Electrical Engineering and Eomputing and lives in Zagreb, Croatia. You can follow him on his blog in English at http://ivoras.net/blog or in Croatian at http://hrblog.ivoras.net/, as well as Google+ at https://plus.google.com/+IvanVoras.

Our courses are available online in Premium Membership.

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 personalized 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