AUUGN

The Journal of AUUG Inc.
Volume 25 • Number 1
March 2004

Features:

KDE 3.2 Review	7
rsynch The best backup system ever	10
Going 3D with Blender: Modelling a chest	12
Tuxpaint: A paint program for kids	15
2003 & Beyond: Final	19
StoreBackup	28
Programmers toolkit: Profiling programsusing gprof	32
Certs for the Masses	34
Cyberinsecurity: The Cost of Monopoly (part 2)	36
Bock Review: Practical VoIP	41
Overheard in the office	42
History of the transport of computer viruses via email	43
Book Review: Just for Fun	45
Book Review: The Complete FreeBSD	45
FreeBSD 5.2 Review	46
Book Review: Perl for Oracle DBAs	49
Book Review: Tomcat - The Definitive Guide	50
Comments of OSS/FS Software Configuration Management Systems	50
Book Review: Free as in Freedom	55

News:

Public Notices	4
AUUG Conference 2004: Call for Papers	18
AUUG: Corporate Members	31
AUUG: Annual Election of Officers	59
AUUG: Chapter Meetings and Contact Details	58

Regulars:

President's Column	3
My Home Network	4

ISSN 1035-7521

Print post approved by Australia Post - PP2391500002

AUUG Membership and General Correspondence

The AUUG Secretary
AUUG Inc
PO Box 7071
Baulkham Hills BC NSW 2153
Telephone: 02 8824 9511
or 1800 625 655 (Toll-Free)
Facsimile: 02 8824 9522
Email: auug@auug.org.au

AUUG Management Committee
Email: auugexec@auug.org.au

President
Greg Lehey
PO Box 460
Echunga, SA, 5153
Bus. Tel (08) 8388 8286, Mobile 0418 838 708, Fax (08) 8388 8725
<Greg.Lehey@auug.org.au>

Vice-president
David Purdue
Sun Microsystems
Level 6, 476 St Kilda Road
Melbourne, Victoria, 3004
Phone: +61 3 9869 6412, Fax: +61 3 9869 6288
<David.Purdue@auug.org.au>

Secretary
Adrian Close
Cybersource Pty.Ltd.
4, 10 Queen Street Melbourne 3000
Business Telephone: +61 3 9621 2377
Business Fax: +61 3 9621 2377
Mobile: +61 417 346 094 <adrian@auug.org.au>

Treasurer
Gordon Hubbard
Custom Technology Australia Pty Ltd
Level 22, 259 George Street, Sydney NSW 2000
Bus Tel: 02 9659 9590, Bus Fax: 02 9659 9510
<Gordon.Hubbard@auug.org.au>

Jonathon Coombes
Cybersite Consulting Pty Ltd
34 Newcastle Road, Wallsend NSW 2287
Business Telephone: +61 2 4965 6989
<Jonathon.Coombes@auug.org.au>

Andrew Frederick Cowie
Operational Dynamics
GPO Box 4339
Sydney, NSW, 2001
Telephone: +61-2-9977-6866
<Andrew.Cowie@auug.org.au>

Steve Landers
Digital Smarties
PO Box 717 Willetton WA 6155
Business phone: +61 8 9313 6868
Business fax: +61 8 9313 6077
<Steve.Landers@auug.org.au>

Stephen Rothwell
IBM Australia Linux Technology Centre
8 Brisbane Ave Barton ACT 2600
Business Telephone: +61 2 62121169
<Stephen.Rothwell@auug.org.au>

Michael Still
Phone: +61 414 382 568 <mikal@auug.org.au>

AUUG Business Manager
Elizabeth Carroll
AUUG Inc
PO Box 7071
Baulkham Hills BC NSW 2153
<busmgr@auug.org.au>

Editorial

Con Zymaris <auugn@auug.org.au>

What price industry development? I've been involved in recent months in various discussions related to the theme of getting the Australian state and federal governments to look at adopting particular types of software. A large part of the emphasis of this exercise is to get the government's procurement policies more closely aligned with efforts to help build Australia's ICT capabilities

Not surprisingly, the 'vehicle' through which the fortunes of both the government's ICT facilities and local ICT industry development can be improved is Free and Open Source Software (FOSS). We'll, that's my working thesis anyway.

The blipvert version of why FOSS can add value here, goes something like this. FOSS, implementing open standards, is a great platform leveller for all consumers of technology. It allows them to freely select the best solution provider for any class of requirement. Economically, this model would lead to a vibrant and competitive marketplace for technologies and services, creating the best environment possible for those consumers.

From an industry development perspective, Open source helps to contain the fears and risks that government procurers see whenever they consider Australian-made software; these risk-averse buyers are filled with dread by the possibility that the Australian supplier will fold and that the source code for the solution will be orphaned. Too many Australian software publishers have done just that in the past, so the fear is somewhat warranted. Solutions by local suppliers, offered under perpetual-access open source licences, mitigate these concerns. It's a simple workaround for a simple roadblock, and it's helping to get local software companies back onto the procurement radar of the federal and state governments..

Great! Now what? Well, why shouldn't Australian governments, help to accelerate this re-introduction of Australian-sourced (or at least supported, in the case of open source,) software through stronger consideration policies? If the case can be made (and proven) to governments, that adopting more of this class of software helps local industries and is about as good a solution from a fitness-for-purpose and value-for-money perspective, then why not push for its broader and deeper adoption or maybe even preference?

I've heard many claim that this isn't in the best interests of government, nor of the open source solution industry which purports to sell this class of products and services: why not let it compete on its own merits? Government is not in the business of picking solutions unless they win in the Darwinian natural selection stakes of value-for-money and fitness-for-purpose ,or so they state.

Oh, but government *is* in the business of bypassing the blind evolutionary pcesses of natural selection, especially if the case can be made that it benefits local industry. It's just that they are not consistent in the application of this policy. Therefore, you will find that government, which is the biggest single buyer of cars in the country, almost always selecting Australian-made vehicles to the exclusion of foreign cars. The government procurement ratios of the two are far removed from the reality of the broader open market, where the sales volume of foreign cars is approaching parity with local product. So, if government can push stronger local supplier consideration for cars, helping local industry in the process, why not with software? Cheers, Con

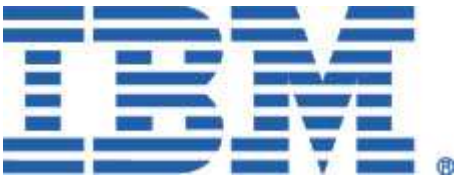
Contribution Deadlines for AUUGN in 2004

Volume 25 • Number 2 – June 2004: **May 15th, 2004**

Volume 25 • Number 3 – March 2004: **August 15th, 2004**

Volume 25 • Number 4 – December 2004: **November 15th, 2004**

AUUG Incorporated gratefully acknowledges
the support of its corporate sponsor:



AUUGN Submission Guidelines

Submission guidelines for AUUGN contributions can be obtained from the AUUG World Wide Web site at:

http://www.auug.org.au/

Alternately, send email to the above correspondence address, requesting a copy.

AUUGN Back Issues

A variety of back issues of AUUGN are still available. For price and availability please contact the AUUG Secretariat, or write to:
AUUG Inc
PO Box 7071
Baulkham Hills BC NSW 2153

Conference Proceedings
A limited number of copies of the Conference Proceedings from previous AUUG Conferences are still available. Contact the AUUG Secretariat for details.

Mailing Lists

Enquiries regarding the purchase of the AUUGN mailing list should be directed to the AUUG Secretariat.

Disclaimer

Opinions expressed by the authors and reviewers are not necessarily those of AUUG Inc., its Journal, or its editorial committee.

Copyright Information

AUUGN is the journal of AUUG Inc., an organisation with the aim of promoting knowledge and understanding of Open Systems, including, but not restricted to, the UNIX® operating system, user interfaces, graphics, networking, programming and development environments and related standards.

President's Column

Greg Lehey <Greg.Lehey@auug.org.au>

I've heard a few comments recently that I'm taking AUUG away from its traditional UNIX focus towards “Open Source”, and that we are neglecting proprietary UNIX. In some cases, I think that “complaint” might be a better word than “comment”.

The concerns are understandable. Last year we were talking about the possibility of merging with Linux Australia, a vehemently Open Source group. This merger hasn't happened yet. It's not from the table. On the other hand, it did give us the impetus to organize the “Linux and Open Source in Government” conference in Adelaide in January (see the report in this issue), in cooperation with members of Linux Australia. It would be easy for members to get the feeling that we're drifting away from our membership basis.

But the concerns are still unfounded. Free software has been a part of UNIX for as long as I can remember. Think of examples like *sendmail* and BIND, which were available free long before the current Linux or Open Source wave started. Nowadays, all big UNIX vendors are looking at and promoting Linux. The only thing that is new is that the public at large have become aware of the idea of free software.

At some point in the last few years, AUUG has woken up and looked around and found that the rest of the world has discovered what we have known for a long time. Some people, particularly those with a Linux background, have talked to me about “AUUG jumping on the Open Source bandwagon”. That's not the case at all: we've been using “open source” all the time. What's different now is that there's a word for it.

But what of the users of proprietary UNIX? Until ten years ago, UNIX *was* proprietary, and so we all used it. Then Linux and the free BSDs appeared, but it took a few years for them to be taken seriously. That has changed: proprietary UNIX is looking decidedly the worse for wear, both in the marketplace and in our membership. We need to embrace the free UNIX variants as well. Looking at the composition of the current AUUG board of directors, we have four people who use BSD (including president, vice-president and secretary) and five who use Linux. That doesn't leave much change out of nine people. To my knowledge, only one member uses proprietary UNIX at all.

This is one of the considerations that has caused us to consider merging with Linux Australia: with one minor detail, the technical interests of our membership are the same. Yes, there are differences in the community, too many to discuss in this column, but the technical interests are pretty much identical.

The one difference is a big one: proprietary UNIX is our *raison d'être*. We would be betraying our principles if we abandoned it, and we have no intention of doing so. In fact, this proves to be the biggest hurdle in any potential merger with Linux Australia: Linux Australia is profoundly (some say “religiously”) Open Source, to the exclusion of any proprietary software. Their attitude to BSD is softening, but only to the extent that it's free. We can't consider merging with a group that refuses to recognize part of our membership base.

It's possible that this composition doesn't represent the users of proprietary UNIX well enough. That would certainly explain the concerns that have been expressed. There's also a simple answer to the problem: elect representatives to the board. Along with this issue you should find an election nomination form. We're electing four officers (President, Vice-President, Secretary and Treasurer) and five “ordinary” board members. We have “job descriptions” on the web site at http://www.auug.org.au/policy/officer-duties.html. Take a look and consider whether one of them isn't for you.

Public Notices

Upcoming Conferences & Events

Windows NT4 Migration to Samba-3, 1-Day Seminar
Instructor: John H. Terpstra

Includes a FREE copy of John H. Terpstra's latest book, "Samba-3 By Example: Practical Exercises to Successful Deployment" rrp $64.95

Sydney, Wednesday 5 May 2004
Location: Vibe North Sydney, 88 Alfred Street, Milsons Point NSW 2061

Melbourne, Wednesday 12 May 2004
Location: Duxton Hotel Melbourne, 328 Flinders Street, Melbourne VIC 3000

Canberra, Tuesday 25 May 2004
Location: Rydges Lakeside Canberra, London Circuit, Canberra ACT 2600

My Home Network (March 2004)

By: Frank Crawford <frank@crawford.emu.id.au>

Welcome to a new year (already a quarter over) and a new column. In recent times I've written about upgrades, changes, etc, but this time Iwill be a bit different, I'll talk about my new toy, and how I set it up.

Late last year, I purchased a laptop for home use, because I wanted to be able to go sit under the trees in the back yard and yet still do work. Amazingly my finance manager (and wife) agreed to all this, and then proceeded to suggest that one of my daughters would find it useful, as she wants to be a writer. As it turns out, I think she was even more devious and was really planning to get me out of the house and into the garden, which she enjoys working on.

Anyway, as it turns out we all agreed and I purchased a new laptop, a Medion MD6100. It is a fairly large system, with a 14.1" TFT, XGA display, with a 2.6GHz Intel Pentium 4 (not the Pentium 4M), DVD ROM/CDRW, etc, etc. And of course it came with Microsoft Windows XP Home edition preinstalled. All in all, it was good value and had just about everything I wanted. However, as with most new laptops, just about every chipset and device is so new that no standard O/S (Linux or Microsoft) have all the drivers and you have to hunt for them.

One good feature of the pre-install is that Medion partitioned the 40Gb disk into three separate partitions, the Windows system partition, a large data partition and an "emergency" partition. This obviously leads to the decision to have a Windows partition, a Linux "partition" and leave the emergency partition untouched. Of course, it wasn't as simple as that, since there was data on the D: drive to save (special driver packages and tools) and the C: drive was too small for regular use.

So, as a first step, copy off all the data on the D: drive, boot up one of the Linux CD distributions (Knoppix 3.1) and away we go. Of course it wasn't that simple, in particular, the X11 graphics driver (`nv') wasn't really correct for the screen, so it was necessary to run every thing in text mode; not a big issue. More of a problem was to resize the partitions. The C: drive, i.e. /dev/hda1 already had an NTFS filesystem, a type that `parted' was not willing to handle, and secondly even if I grew the partition, I was still stuck with growing the filesystem. Anyway, after a bit of hacking and abuse, I did manage to get the disk partition laid out as I wanted, i.e. /dev/hda1 of 19Gb, /dev/hda2 of 19Gb and /dev/hda5 (the recovery partition) unchanged at 500Mb. I haven't really gone into the disk layout, but /dev/hda2 was already an extended partition, and all I really did was delete one of the logical partitions within it, and then resize the partition.

Of course, Windows XP doesn't have any utility to change the filesystem size, as they expect you to purchase a commercial product. If you've read my previous columns, you would know that there are a lot of open source utilities for working on NTFS filesystems. In particular, the Linux NTFS driver site has a tool for resizing NTFS partitions, called `ntfsresize' (see http://mlf.linux.rulez.org/mlf/ezaz/ntfsresize.html). Apparently all it does is update a couple of entries and then allows Windows `chkdsk' to do the rest of the work on the next reboot. Okay, first step done, i.e. repartitioned the system with no loss of any of the Windows

data.

Onto the next step, Linux installation. If you read my previous column, you would know that I am moving things to Fedora Core 1 (in fact now all my systems run Fedora, Red Hat 9 is long gone), so obvious first step was to insert the CD and run. In general, this worked fairly simply, except again for problems with the `nv' driver in Xfree86. This just forced me to perform the install in text mode. For those interested theproblem wasn't that `nv' wouldn't work, rather, the work area wouldn't fit on the screen, so it was always panning, and there was a lot of ghosting, all of which made it almost unusable. The basic problem was that the nVidia chipset was not really supported by the `nv' driver, but was close enough to almost work.

I also was stuck for a little while with CD "error" part way through the installation, despite previously checksumming the CD contents at the start. A bit of googling and I quickly turned up some notes from Alan Cox regarding the problem (http://fedora.artoo.net/faq/). Put simply, I needed to enable DMA in the installation kernel for the DVD drive. So, I added the right magic (i.e. booting with "linux allowcddma") and away it went.

Anyway, after an hour or so, it was installed, but that was only the start. I didn't really want to stay in text mode forever, and so I went hunting. Again google is your friend, and this time I turned up the TuxMobile site (http://www.tuxmobile.org), which has a number of details for the installation of various laptops and other mobile linux platforms. In fact there were a few descriptions of the installation of SuSE on a Medion MD6100. Unfortunately, they were in German and Google's translation program kept stopping halfway through. Oh well, back to my traditional method of translating other languages, "gee that word is like the English for X, lets assume it is" or "that technical term/URL/... is the same in both languages", etc.

The first step I did was compare my hardware to their list. To do this, I just ran `/sbin/lspci' since just about everything is PCI based. This is what it came out with:

```
00:00.0 Host bridge: Silicon Integrated Systems
[SiS] SiS 645xx (rev 03)
00:01.0 PCI bridge: Silicon Integrated Systems
[SiS] SG86C202
00:02.0 ISA bridge: Silicon Integrated Systems
[SiS] SiS85C503/5513 (LPC Bridge) (rev 14)
00:02.3 FireWire (IEEE 1394): Silicon Integrated
Systems [SiS] FireWire Controller
00:02.5 IDE interface: Silicon Integrated Systems
[SiS] 5513 [IDE]
00:02.6 Modem: Silicon Integrated Systems [SiS]
AC'97 Modem Controller (rev a0)
00:02.7 Multimedia audio controller: Silicon
Integrated Systems [SiS] Sound Controller (rev a0)
00:03.0 USB Controller: Silicon Integrated Systems
[SiS] USB 1.0 Controller (rev 0f)
00:03.1 USB Controller: Silicon Integrated Systems
[SiS] USB 1.0 Controller (rev 0f)
00:03.2 USB Controller: Silicon Integrated Systems
[SiS] USB 1.0 Controller (rev 0f)
00:03.3 USB Controller: Silicon Integrated Systems
[SiS] USB 2.0 Controller
00:04.0 Ethernet controller: Silicon Integrated
Systems [SiS] SiS900 PCI Fast Ethernet (rev 90)
00:0c.0 CardBus bridge: ENE Technology Inc CB1410
Cardbus Controller
01:00.0 VGA compatible controller: nVidia
Corporation: Unknown device 0187 (rev a2)
```

The second was to go through the start up log ("/var/log/messages") and see what was known and what wasn't. Finally go back to my German translation and pull out the relevant technical information. Ohh, and look at the other documentation that came with the system and even the box it was packed in (you'll be surprised what you can learn from the blurbs on the box).

From this it was immediately obvious that I needed to get the latest nVidia driver for Xfree86. Off to http://www.nvidia.com and download their latest driver (which I see lists the PCI id's and my system is a GeForce4 488 Go), install and watch it not work. Hmm, back to more manual reading and documentation. Again reading the box, the advertising makes a big play of the new 8X AGP performance, try some tweaking of the AGP setting for the nVidia driver and away we go. In fact what I needed to do was disable the Linux Kernel AGP driver (which was a module anyway) and allow the nVidia `nvidia' X11 driver to manage the AGP directly. At that point it was running. As a little side note, the original nVidia driver did not let me switch between graphics and text mode, it looks like even they got it wrong. Later version now do have it working, so on shutdown, I now see what is happening.

Next item on the list, shutting down the hardware! The Medion MD6100 doesn't support APM, but rather only support ACPI, and while this is far more extensive a control mechanism, not all the tools come with the base installation. There are two aspects to support ACPI, the first is within the kernel, which has specific ACPI modules, and the second is a daemon to monitor the status of the various items.

Obviously, the most important one for a laptop is if the system is on AC or battery, and the status of the battery. Additional sensors give CPU temperature and status, and even whether the lid is open or closed. All of this is covered by the acpid daemon, in an rpm distributed with Fedora. In fact the kernel supplied with Fedora is compiled with ACPI modules, although by default, they are disabled.

The acpid daemon consists of a number of separate configuration files, in /etc/acpi/events, which are matched for any acpi event and then run the specified action. For example, the supplied sample file (sample.conf) is:

```
# This is a sample ACPID configuration

event=button/power.*
action=/sbin/shutdown -h now
```

which obviously shuts down the system when the power button event is pressed. More importantly, I have also added events for changing to/from the ac adapter (more on this later). Eventually it would be possible to put the system into hibernation on closing the lid, I just need to add the relevant kernel patches (or run Linux 2.6).

Before going about the acpid daemon, another interesting feature that is included in the Fedora kernel, and also available in the standard Linux kernel after 2.4.23 is a "laptop_mode" which batches kernel disk I/O, allowing the disk drive to become idle long enough for power-saving features to take effect. This is enabled by "echo 1 > /proc/sys/vm/laptop_mode", but really needs additional kernel options set. These additional steps are performed by the default script for the apm daemon, but no scripts are provided for acpid. Searching the net for further details, the main other item is to change how often disk syncs are being performed though `/proc/sys/vm/bdflush' (see an example below). To enable laptop_mode, the following script (/etc/init.d/laptop_mode) is available:

```
#!/bin/sh
#
# start of stop laptop mode, best run by a power
management daemon when
# ac gets connected/disconnected from a laptop
```

```
#
# FIXME: assumes HZ == 100

# age time, in seconds. should be put into a
sysconfig file
MAX_AGE=600

# kernel default dirty buffer age
DEF_AGE=30
DEF_UPDATE=5

if [ ! -w /proc/sys/vm/laptop_mode ]; then
        echo "Kernel is not patched with
laptop_mode patch"
        exit 1
fi

case "$1" in
        start)
                AGE=$((100*$MAX_AGE))
                echo -n "Starting laptop mode"
                echo "1" > /proc/sys/vm/laptop_mode
                echo "30 500 0 0 $AGE $AGE 60 20 0"
> /proc/sys/vm/bdflush
                echo "."
                ;;
        stop)
                U_AGE=$((100*$DEF_UPDATE))
                B_AGE=$((100*$DEF_AGE))
                echo -n "Stopping laptop mode"
                echo "0" > /proc/sys/vm/laptop_mode
                echo "30 500 0 0 $U_AGE $B_AGE 60 20
0" > /proc/sys/vm/bdflush
                echo "."
                ;;
        *)
                echo "$0 {start|stop}"
                ;;

esac

exit 0
```

and then through acpid add the following event and action scripts:

```
# This rule defines details for the battery

event=battery.*
action=/etc/acpi/actions/battery "%e"
```

and

```
#! /bin/sh
#
# Handle change of ac power state
#
# Taken from /etc/sysconfig/apm-scripts/apmscript

LANG="C"
export NOLOCALE=1
# May as well read from a standard location
[ -e /etc/sysconfig/apmd ] && . /etc/sysconfig/apmd

set -- $*
BATTERY="${1:-battery}/${2:-BAT0}"

critical () {
    LC_ALL=C grep -q critical-line /
proc/acpi/$BATTERY/state &>/dev/null
}

if critical; then
    # Battery low. If you want to be on the safe
side, maybe put
    # the harddisk into extreme powersaving, or
"apm -s" here.
    if [ -n "$LOWPOWER_SERVICES" ]; then
        [ -d /var/run/apmd ] || mkdir -p /
var/run/apmd
        touch /var/run/apmd/LOW_POWER
        for i in $LOWPOWER_SERVICES; do
            /sbin/service $i stop
        done
    fi
fi

exit 0
```

(This script is taken from the apm script and uses the same configuration file and values.)

Staying on the kernel track there are two other related modules needed. The first is fairly common, the ability to mount NTFS file systems (to allow file interchange with Windows XP). The standard Fedora kernel doesn't include NTFS, but there are two methods to enable it. Firstly, roll your own kernel from the standard Linux kernel source. This is what I did, and allows the addition of other features. The other method, which I recently came across was that the Linux NTFS kernel team have compiled up installable modules for all distributed Fedora kernels (as well as Red Hat, etc). These can be found at http://linux-ntfs.sourceforge.net/rpm/index.html.

The second item to add to the kernel, needs further explanation. The Medion MD6100 comes with what is known as a WinModem, i.e. chip set that has some modem functionality, but requires a kernel driver to do most of the work. It is cheap and nasty for the PC builders, tends to work with Windows, but usually useless for non-Windows systems. However, as with most such devices, someone in the Open Source movement is working on it, and in fact they now claim to have a driver available for Linux. In fact the company doing the development is Smart Link who are the chipset manufactures and also develop drivers for Windows. Of course the Linux driver is unsupported, and does have a binary only component, which is not covered by the GPL, but does seem to work.

So with this, and appropriate configuration, every device is available to Linux, and seems to be better supported than Windows. I haven't really mentioned details such as how to configure the builtin Ethernet port, USB ports, IR port, Firewire interface and PCMCIA/Cardbus adapter (supported by the `yenta_socket' module). The modules compiled up for the system are given in "/etc/modules.conf", shown below:

```
alias usb-controller usb-ohci
alias usb-controller1 ehci-hcd
alias ieee1394-controller ohci1394
alias eth0 sis900
alias sound-slot-0 i810_audio
post-install sound-slot-0 /bin/aumix-minimal -f /
etc/.aumixrc -L >/dev/null 2>&1 || :
pre-remove sound-slot-0 /bin/aumix-minimal -f /
etc/.aumixrc -S >/dev/null 2>&1 || :
alias eth1 orinoco_cs

# IrDA over a normal serial port, or a serial port
compatible IrDA port (SIR)
alias tty-ldisc-11 irtty

# IrCOMM (for printing, PPP, Minicom etc)
alias char-major-161 ircomm-tty # if you want
IrCOMM support

# IRLAN
# But currently the IrLAN protocol is no longer
maintained
# by the Linux/IrDA core team.
alias irlan0 irlan

# To be able to attach some serial dongles
# These values are hard-coded in irattach (not
instance order)
alias irda-dongle-0 tekram # Tekram IrMate IR-210B
alias irda-dongle-1 esi # ESI JetEye
alias irda-dongle-2 actisys # Actisys IR-220L
alias irda-dongle-3 actisys # Actisys IR-220L+
alias irda-dongle-4 girbil # Greenwich GIrBIL
alias irda-dongle-5 litelink # Parallax
LiteLink/ESI JetEye
alias irda-dongle-6 airport # Adaptec Airport 1000
and 2000
alias irda-dongle-7 old_belkin # Belkin (old)
SmartBeam dongle
```

```
alias irda-dongle-8 ep7211_ir # Cirrus Logic EP7211
Processor (ARM)
alias irda-dongle-9 mcp2120 # MCP2120 (Microchip)
based
alias irda-dongle-10 act200l # ACTiSYS Ir-200L
alias irda-dongle-11 ma600 # Mobile Action ma600

# To use the FIR driver. This applies only to the
specific device!!!

#options nsc-ircc dongle_id=0x09     # NSC driver
on a IBM Thinkpad laptop
#options nsc-ircc dongle_id=0x08     # HP Omnibook
6000
#alias irda0 nsc-ircc

# options smc-ircc ircc_irq= ircc_dma=
# alias irda0 smc-ircc

# options toshoboe max_baud=
# alias irda0 toshoboe

# options w83977af_ir io= io2= irq= qos_mtt_bits=
# alias irda0 w83977af_ir

# IrNET module...
alias char-major-10-187 irnet # Official allocation
of IrNET

# WinModem modules ...
alias char-major-212 slamr
alias char-major-213 slusb

#below nvidia agpgart
alias char-major-195 nvidia
```

Another issue, over and above the kernel, is that X11 requires some configuration, including the use of the TouchPad for use in multiple fashions. Rather than just being seen as a simple 3-button mouse, it has a number of extra functions, and which can be used through the Synaptics linux driver (http://tuxmobil.org/touchpad_driver.html). The definition for this is added to the XF86Config file, and add such features as horizontal and vertical scrolling, by just sliding your finger along the edge of the pad. (However, after fixing it up, I found for long use on the system, it is easier to use a little USB mouse, specifically designed for laptops!)

Finally, one other useful little item is `lineak' which allows keyboard to be configured as multimedia, easy access and Internet keys, much as found in many Windows systems. This tool can be found at http://lineak.sourceforge.net, and while it still seems to have bugs, mainly due to compilation and creation under a different Linux platform, it still seems to work well. The only thing I can't currently use is the OSD (On-Screen Display) feature to inform me what is being activated. This isn't a big loss, just a niggling issue.

Probably the last feature of note is the DVD ROM/CDRW combo device, which is easily handled by X-CDRoast, although this isn't reflected in /etc/modules.conf given above.

So just to show you how useful this is, I have spent time sitting in the garden working on the laptop. Even today, I actually wrote much of this article sitting in a hotel between presentations of an ITIL conference. It certainly made this work much, much easier. So, what do you think, are all these details useful, or am I just filling space in AUUGN? Let me know or give me some new things to write about. Have fun!

KDE 3.2

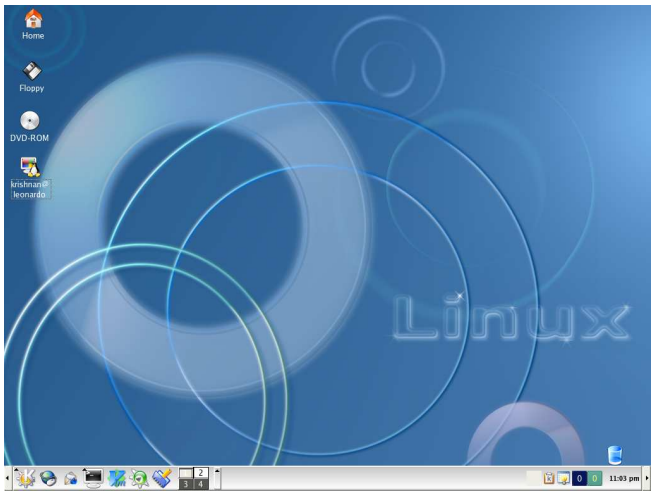
Author: Krishnan Subramanian <krishnan@fedoranews.org>

Today I installed KDE 3.2, third major release of award winning KDE3 desktop platform, on my Fedora box. I have been using KDE 3.2 RC for the past few days and the final version from today. My first impression is "wow".

KDE 3.2 provides an integrated desktop along with various applications to carry out common desktop tasks such as web browsing, email, instant messaging, multimedia, graphics, etc. Some of the impressive features which you will notice include

- Increase in speed evident from faster application startup time
- Improvements in usability and performance
- Better appearance through interface refinement
- Browser performance boost evident through better webpage rendering

Upgrading to KDE 3.2 is a breeze. If you are a newbie and want to learn how to do it, you can refer to my HOWTO. I started my installation and within few minutes I am logged into my new KDE 3.2 desktop.



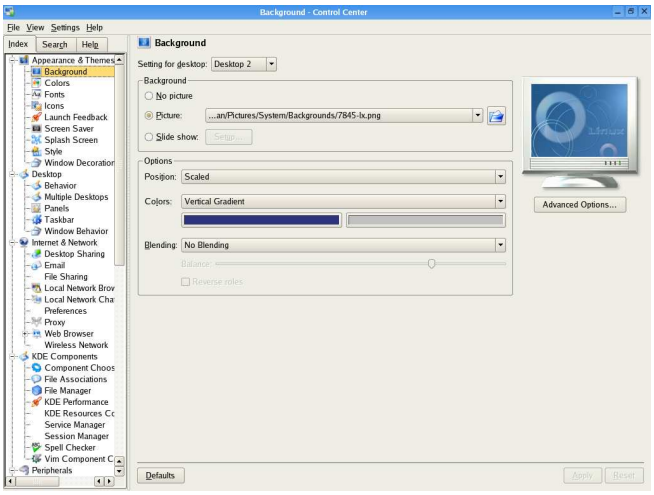
The desktop is very polished and you can configure it in any way you want by right clicking on the desktop. You can setup your desktop background as a slide show so that the background picture changes at predetermined intervals. The style and window decorations are very refined increasing the overall appearance. I love plastik for style and window decoration. A better icon set is also available. Now that you can find a wide array of themes and icon sets in www.kde-look.org, you can customize your KDE desktop in any way you want. In fact, you can even select the KDE splash screen (which appears when you login) from the available choices.

The K Menu is better organized now. It is grouped into "Most Used Application", "All Applications" and "Actions". Even the applications are grouped in a much better way compared to earlier version.

The new KHotkey feature is really hot. You can create keyboard shortcuts and mouse gestures for various tasks. This comes very handy. People used to such features in Microsoft Windows environment will love this feature. It is really cool to press the "Windows" key in your keyboard and see KMenu pop up in your

screen.

The control center is well spruced up and better structured in KDE 3.2. Some of the tabs like background, window decoration, style etc. are redesigned.



Some of the welcome addtions to control center are

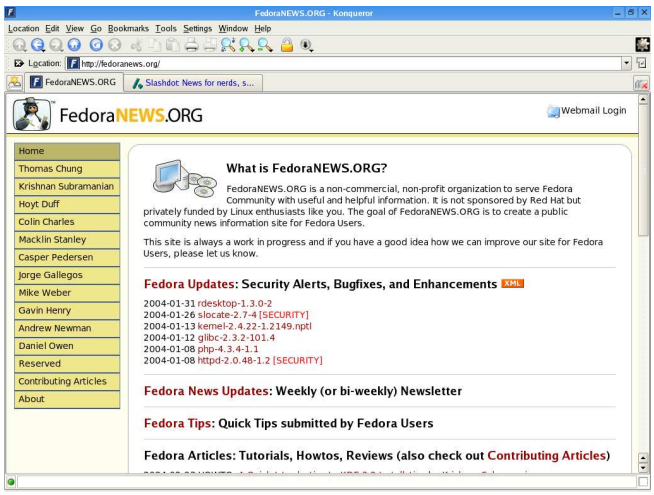
- Splash Screen - where you can select a KDE splash screen of your choice
- Wireless Network - where you can configure your wireless network. You can save upto four different configurations.
- Vim Component Configuration - where you can configure Vim to use inside KDE
- KHotkeys - where you can specify keyboard shortkeys and mouse gestures to lauch applications in KDE
- KDE Wallet - where you can configure KDE Wallet to store your internet and local passwords
- Sony Vaio Laptop - where you can configure the hardware for this laptop

KDE 3.2 has more countries under Country/Region. Also these countries are better organized. This is a very positive step in the internationalization efforts of KDE.

Another welcome feature in the control panel is the "Font installer". With this, installation of new fonts is a breeze. This is very useful for people who want to install their regional fonts and other extra fonts (many fonts are available in kde-look.org). The best aspect of the font installer is the instant preview available with it. I feel this is one of the greatest additions to KDE.

Many new applications are added and some of the existing applications have been upgraded. It is quite impossible to discuss all the applications available in KDE 3.2. I will just discuss some of the applications based on my preferences.

Konqueror: This is the central part of KDE environment. it is a web browser, file manager, network browser and so on. Konqueror has finally matured as a web browser. I feel, though many would disagree with me, that rendering of sites is sometimes better than Mozilla. I find this difference while checking out IE based sites. This is just my observation and I cannot quantify this in any way.

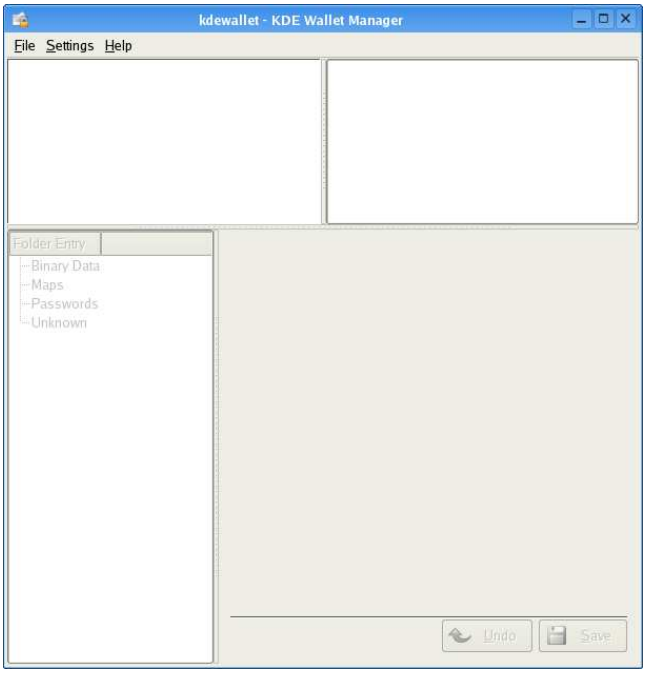


Konqueror now supports tabs for both web browsing and file management. This is very useful if you don't want clutter on your desktop or switch between various folders/webpages often. Konqueror also comes with a universal sidebar with lots of functionality. One of the best features of Konqueror is the addition of service menu. You can add items to your right click through service menus. There is a nice tutorial for this. Overall it gives you a better browsing experience.

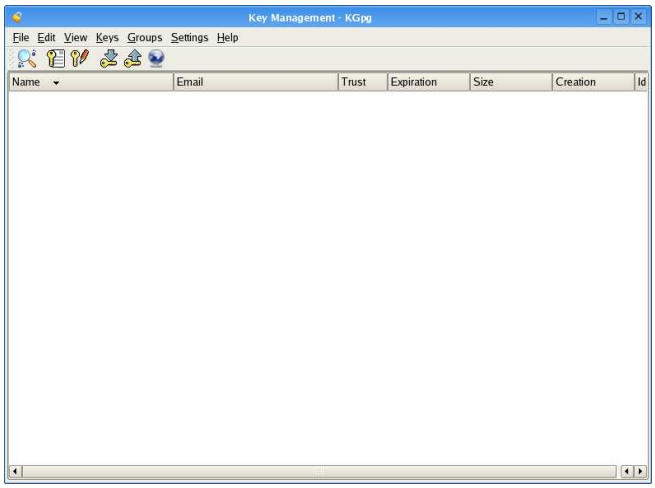
Editors: KDE 3.2 comes with three editors. They are Kate, KEdit and KWrite. I don't understand the need for three editors. Kate and KWrite are very goood editors with many features but KWrite takes a long time to load.

There are so many other welcome additions in this version of KDE. Quanta now has a WYSIWYG Web development environment. There is a new release of KDevelop added to KDE 3.2 and so on. Now let me turn my attention to some of the new applications. Some of them like KWallet and KGpg are great additions. These applications will play a very important role in your daily desktop usage.

KWallet: This provides an integrated secure storage of passwords and web form data. This works very well with KDE applications like Konqueror, Kopete etc. Each program can be given different level of access. This docks nicely into KDE panel. You can shut it off after a specified time period or as soon as screensaver starts. You will definitely find KWallet very handy in your internet browsing experience.

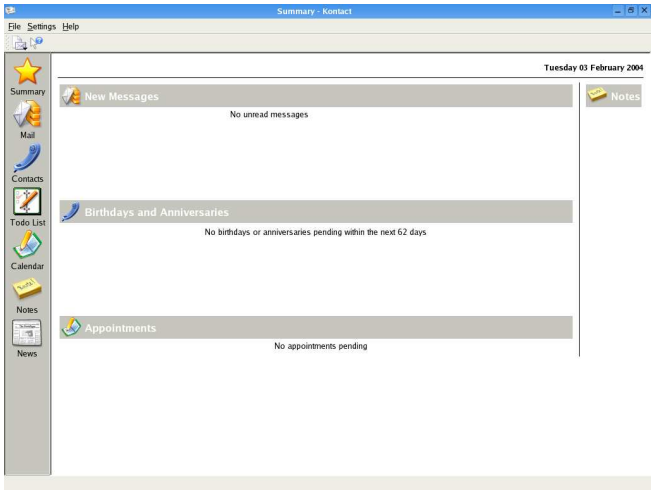


KGpg: This is a key manager which can be used to import, export, delete, sign, generate and edit keys. This is integrated with Konqueror very well. There is support for support for symetric encryption. Multiple keys & default key encryption. People who use GPG keys will find this utility very handy.



Kopete: KDE 3.2 comes with its own instant messaging software. This is an instant messenger with support for AOL Instant Messenger, MSN, Yahoo Messenger, ICQ, Gadu-Gadu, Jabber, IRC, SMS and WinPopup. With Kopete, you need not use separate clients for different instant messaging network. This docks nicely into KDE panel.

Kontact: This combines KAddressbook, KMail, KNode, KOrganizer, KWeather, and KNotes under a common GUI with a sidebar to select different modules. This is similar to Ximian Evolution or Microsoft Outlook. Unlike MS Outlook, you can run the modules separately or as an all in one application. You can add/remove modules depending on what you want. This is definitely a good start for KDE people. This one was long overdue and I hope it matures into an excellant piece of software with time.



There are so many aspects of KDE 3.2 I haven't reviewed in this article. This does not mean that they are not important. I once again want to emphasize that the selection of applications for review is based on just my preferences. To sum it up, with so many enhancements, an upgrade is worth your time. Applications like KWallet and KGpg are absolute necessities. I would like to point out that I have written this review after just a day's experience with KDE 3.2 (although I have been using KDE 3.2 RC1 for sometime now) and I haven't encountered any problems right now. I would like to hear comments from readers about their KDE 3.2 experience so that I can update this article at a later stage. Send me an email about your comments.

Just go ahead and install KDE 3.2.

This article is re-printed with permission. The originals can be found at:

http://fedoranews.org/krishnan/review/kde3.2/

rsync: The Best Backup System Ever

Author: Brian Hone <bhone@eink.com>

Abstract

Backup is one of the hardest and most neglected parts of system administration. It is also one of the most important. It is the last defense against hardware failures, security breaches, and the biggest threat of all: end users. While there are many backup systems out there costing many thousands of dollars, which archive to expensive tape drives using buggy proprietary software, there is a better way: Rsync and a cheap disk array.

The Problem

I can give you a long list of reasons why backup is a system administrator's nightmare. If you're a system administrator, though, I probably don't need to. Some of those reasons are: expensive hardware which is broken more often that it is operational, expensive software which is a management nightmare, and long hours spent restoring multiple versions of files. To make matters worse, there is usually very little corporate priority placed on backups, until that inevitable day when they're needed. If you've done backup/restore, odds are you've had this conversation:

User: "I lost a file. I need you to get it back right away."
SysAdmin: "Ok, what's it called?"
User: "I don't know, I think it had an 'e' in the name."
SysAdmin: "Ok, what directory was it in?"
User: "I don't know, it could be in one of these three..."
SysAdmin: "*Sigh* Do you know what date you last used the file?"
User: "Well....I think it was a thursday in either February or April. What's the problem? I thought you people had a backup system to take care of this kind of thing."

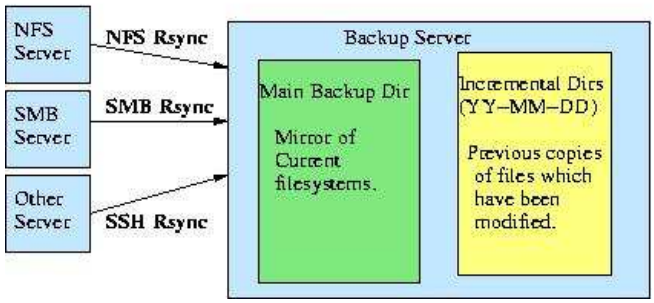
The Rsync Alternative

Rsync is a powerful implementation of a beautiful little algorithm. Its primary power is the ability to effeciently mirror a filesystem. Using rsync, it is easy to set up a system which will keep an up to date copy of a filesystem using a flexible array of network protocols, such as nfs, smb or ssh. The second feature of rsync which this backup system exploits is its ability to archive old copies of files which have been changed or deleted. There are far too many features of rsync to consider in this article. I strongly recommend that you read up on it at rsync.samba.org.

The System

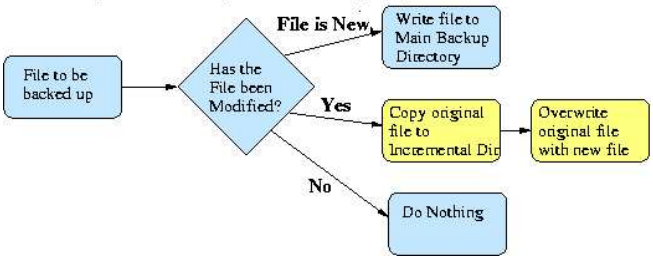
In brief, this system uses a cheap Linux box with a lot of cheap disk and a small shell script which calls rsync. **[Fig 1]** When doing a backup, we tell rsync to create a directory named 'YY-DD-MM' as a place to store incremental changes. Next, rsync examines the servers we backup for changes. If a file has changed, it copies the old version to the incremental directory, and then overwrites the file in the main backup directory. **[Fig 2]**

Figure 1: Structure of an Rsync Backup Server



In general, a day's changes tend to be between a small percentage of the total filesystem. I find the typical average size to be between . 5% and 1%. Therefore, with a set of backup disks which is twice the size of our fileservers, you can keep 50-100 days of incremental backups on hard drive. When the disk becomes full, just swap in a new set of disks, and move the old ones offsite. In practice, it is possible to keep over six months of incrementals on disk. In fact, if you can find space somewhere, you can copy your incrementals to another server before rotating the disks. In this way, you can keep an arbitrarily large number of incrementals on disk.

Figure 2: Backup of a file using Rsync



The Advantages: Disaster Recover and File Restoration Made Easy

Go back to the imaginary conversation above. Now, instead of a cumbersome tape-based system, imagine having six months of incremental backups happily waiting for you on your Linux box. Using your favorite combination of locate/find/grep, you can find all occurances of files owned by our imaginary user, which contain an 'e' and are timestamped on a thursday in February or April, and dump them into a directory in the user's home directory. The problem of figuring out which version is the correct one has just become my favorite kind of problem: someone else's.
Next, imagine our favorite scenario - complete failure. Lets say you have a big nfs/samba server which you lose. Well, if you've backed up your samba configs, you can bring your backup server up as a read-only replacement in minutes. Let's see you try that with tape.

How Rsync/Hard Drive Backup Stacks up Against Tape

	Tape Backup	*Rsync*
Cost	Very High	Low
Full Backup	Fast	Fast
Incremental Backup	Fast	Fast
Full Restore	Very Slow, probably multiple tapes	Fast - it's all on disk
File Restore	Slow, maybe multiple tapes, often hard to find correct version.	Very Fast - it's all on disk and you have the full power of UN*X search tools like find, grep and locate
Complete Failure	Only option is full restore	Can be turned on as a fileserver in a pinch.

The Tools

There are a lot of ways to set this up. All the tools here are open-source, included in standard distributions, and very flexible. Here, we describe one possible setup, but it is far from the only way.

- **The Server:** I use RedHat Linux. Any distribution should work, as should any UN*X. (I've even set this up with Mac OS X) One caveat: a lot of RAM helps.
- **Disk:** The easiest way we've found of building a big cheap set of disk is a PCI firewire card connected to a bunch of cheap IDE disks in external firewire cases. Setting up Linux to use these as one big RAID partition is fairly painless.
- **The Software:** Rsync is a great tool. It is sort of a jacknife of filesystem mirroring. If you don't know about it, check it out at rsync.samba.org.
- **Connecting to Fileservers:** Rsync is very flexible. We use nfs and smbfs. You can also use rsync's own network protocol by running an rsync daemon on the fileserver. You can also tell rsync to use ssh for securely backing up remote sites. See the resources below for information of setting up these connections.

Scripting It

The basic form of this script came from the rsync website. There is really only one command:

```
rsync --force --ignore-errors --delete --delete-
excluded --exclude-from=exclude_file --backup --
backup-dir=`date +%Y-%m-%d` -av
```

The key options here are:

- *--backup*: create backups of files before overwriting them
- *--backup-dir=`date +%Y-%m-%d`*: create a backup directory for those backups which will look like this: 2002-08-15
- *-av*: archive mode and verbose mode.

The following script can be run every night using Linux's built in cron facility. To start the script at 11pm each night, use the command "crontab -e", and then type the following:

```
0 23 * * * /path/to/your/script
```

The Script

Here's my shell script to tie it all together. Again, there are a lot of ways of doing this. This is just one implementation.

```
#!/bin/sh

####################################################
# Script to do incremental rsync backups
# Adapted from script found on the rsync.samba.org
# Brian Hone 3/24/2002
# This script is freely distributed under the GPL
####################################################

###################################
# Configure These Options
###################################

###################################
# mail address for status updates
#  - This is used to email you a status report
###################################
MAILADDR=your_mail_address_here

###################################
# HOSTNAME
#  - This is also used for reporting
###################################
HOSTNAME=your_hostname_here

###################################
# directory to backup
# - This is the path to the directory you want to
# archive
###################################
BACKUPDIR=directory_you_want_to_backup

###################################
# excludes file - contains one wildcard pattern
# per line of files to exclude
#  - This is a rsync exclude file.  See the
#    rsync man page and/or the
#    example_exclude_file
###################################
EXCLUDES=example_exclude_file

###################################
# root directory to for backup stuff
###################################
ARCHIVEROOT=directory_to_backup_to

########################################
# From here on out, you probably don't  #
#   want to change anything unless you  #
#   know what you're doing.             #
########################################

# directory which holds our current datastore
CURRENT=main

# directory which we save incremental changes to
INCREMENTDIR=`date +%Y-%m-%d`

# options to pass to rsync
OPTIONS="--force --ignore-errors --delete \
 --delete-excluded \
 --exclude-from=$EXCLUDES --backup \
 --backup-dir=$ARCHIVEROOT/$INCREMENTDIR -av"

export PATH=$PATH:/bin:/usr/bin:/usr/local/bin

# make sure our backup tree exists
install -d $ARCHIVEROOT/$CURRENT

# our actual rsyncing function
do_rsync()
{
   rsync $OPTIONS $BACKUPDIR $ARCHIVEROOT/$CURRENT
}

# our post rsync accounting function
do_accounting()
{
   echo "Backup Accounting for Day $INCREMENTDIR \
```

```
      on $HOSTNAME:">/tmp/rsync_script_tmpfile
   echo >> /tmp/rsync_script_tmpfile
   echo "####################################">> \
   /tmp/rsync_script_tmpfile
   du -s $ARCHIVEROOT/* >>/tmp/rsync_script_tmpfile
   echo "Mail $MAILADDR -s $HOSTNAME Backup \
      Report < /tmp/rsync_script_tmpfile"
   Mail $MAILADDR -s $HOSTNAME Backup Report < \
      /tmp/rsync_script_tmpfile
   echo "rm /tmp/rsync_script_tmpfile"
   rm /tmp/rsync_script_tmpfile
}

# some error handling and/or run our backup and
# accounting
if [ -f $EXCLUDES ]; then
        if [ -d $BACKUPDIR ]; then
                # now the actual transfer
                do_rsync && do_accounting
        else
                echo "cant find $BACKUPDIR"; exit
        fi
        else
                echo "cant find $EXCLUDES"; exit
fi
```

Resources

- Rsync: http://rsync.samba.org
- NFS: http://nfs.sourceforge.net/nfs-howto
- SMBFS: http://samba.org
- Linux RAID: http://linas.org/linux/raid.html

...

This article is re-printed with permission. The originals can be found at:
http://www.linuxfocus.org/English/March2004/article326.shtml

Going 3D with Blender: Modeling a chest

Author: Katja Socher <katja@linuxfocus.org>

Introduction

In this article we model a chest with Blender.

Modeling a chest

Look at the image above and you see the chest we are going to create. For that we open the stage environment that we built in my first article about Blender. If you haven't read the article and built the stage environment yourself you should have a look at “Going 3D with Blender: Very first steps” first before you proceed with this article. With this stage environment we have kind of a default setting with lights where we can place the chest in.

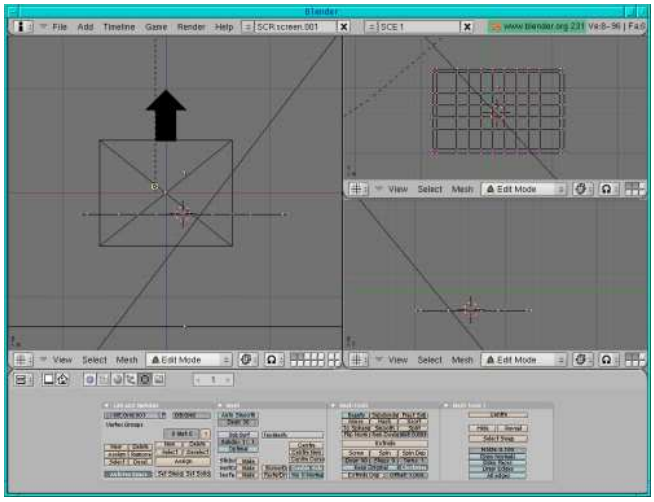
By the way the current version of Blender as of this writing is now 2.31a. The interface of Blender has changed a lot but after working with it for a little while you will find that the changes are really for good. Congratulations and thanks to the Blender team for their excellent job! :)

The shapes

If you look at the chest you can easily see that the two main shapes are simply a box and a cylinder that is cut in half. The difficulty lies in giving it the impression of thickness. The method I will describe heavily uses extrusion. If you have any other suggestions on how to model the chest let me know!

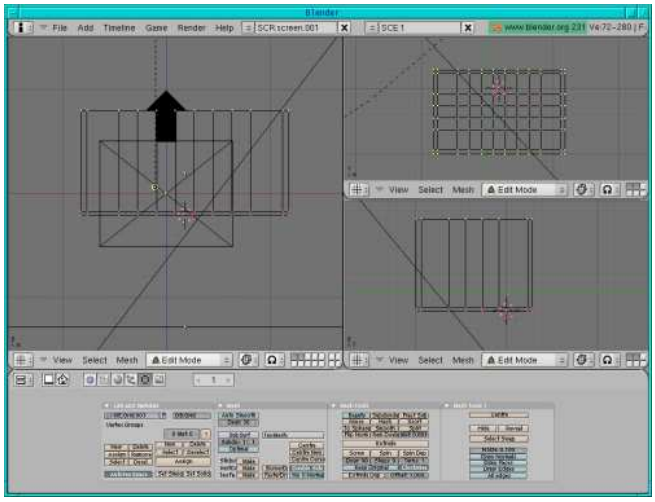
The chest box

For the box add a grid with xres=12 and yres=8 (Space-->Add-->Mesh-->Grid with xres=12 and yres=8) in top view (in layer 2). In the second inside rectangle select the single lines and move them out close to the outer rectangle: Start with the second line from the left and select it. Next press g and then move it with the arrow key to the left. Do the same with the other three second lines of the rectangle.



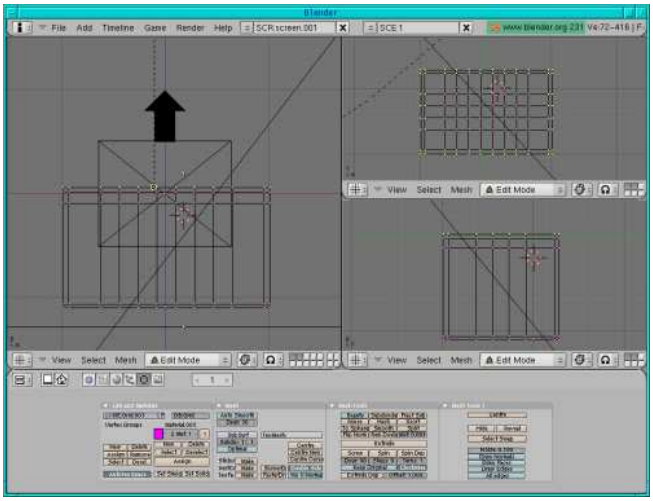
Now select everything (press a twice) and in front view extrude a bit (press e, enter, arrow key, enter) so that you get a board.

In top view select the two outer rectangles (deselect the points inside by pressing b and the right mouse button)and in front view extrude them and move them up (press e, enter, arrow key, enter).



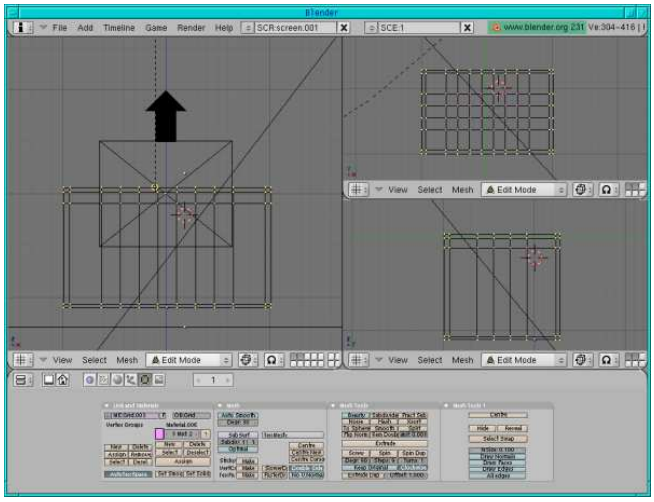
A simple model of the box is ready.

Select the top line in front view (if it isn't selected anylonger) and extrude a bit (a bit more than the distance between the two lines on the bottom), then press e again and extrude again a bit (as much as the distance at the bottom).



Assign the whole box a pink colour (go to the material buttons, press "add new" and move the colour sliders to R=1,G=0,B=1), then select the inside of the box (that's everything except the four lines on the corners in top view and the bottom line in front view) and assign a light pink colour (in the edit buttons press "New", "Select", then go to the material buttons, press "add new" and move the colour sliders to R=1,G=0.6,B=1, go back to the edit buttons and press "Assign").

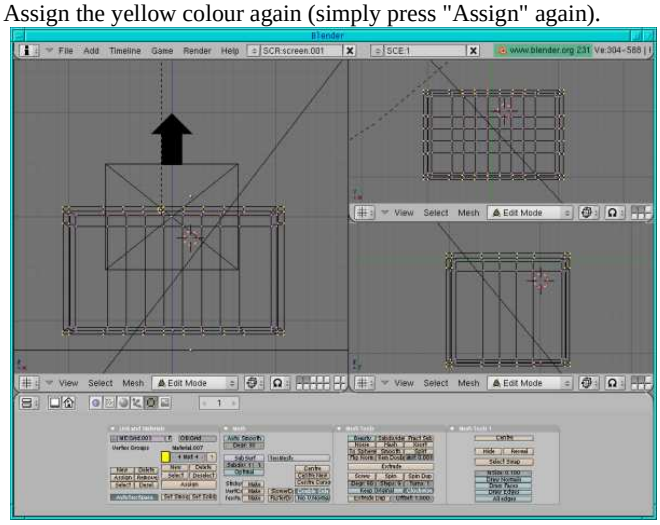
Now select all points (press a twice), then deselect (press b and the right mouse button) the inner points in top, side and front view each (see figure below).



Assign a yellow colour (in the edit buttons press "New", "Select", then go to the material buttons, press "add new" and move the colour sliders to R=1,G=0.1,B=0, go back to the edit buttons and press "Assign").

In top view extrude and scale up (press e, enter, s, arrow key, enter).

Assign the yellow colour again (simply press "Assign" again).



The box is ready.

THE HANDLES

On the two smaller sides we will add some handles: In front view hit Space, Add-->Mesh-->UV Sphere with the Segments and Rings at 32 (which usually is the default value), scale it down (press s) and flatten the sphere a bit (press s and by holding the middle mouse button down restrain the scaling down to the thickness). Move it so that it is on the corner of the side in front view and in the middle of the box in side view. Go to the edit buttons and press smooth, give it a light pink colour (R=1,G=0.6, B=1). Copy it (shift + d) and move (press g) it to the other side.

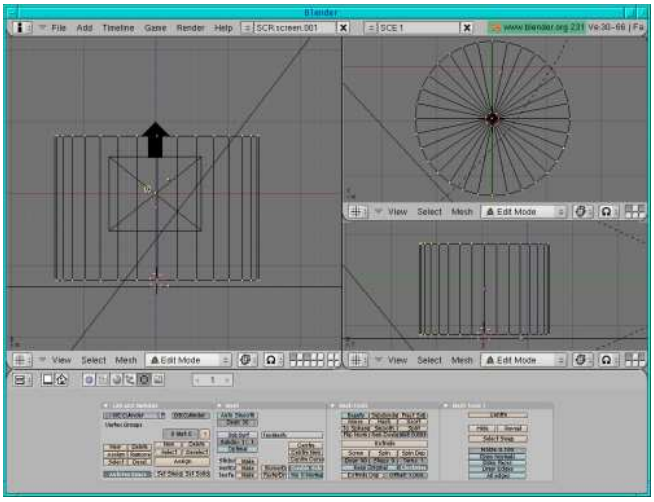
For the ring add a mesh circle (hit Space, Add-->Mesh-->Circle (Vertices=32)) in side view, then scale it down, then in side view (in edit mode) press e and then s and scale it up for the right thickness. Select all points (press a twice) and now extrude (press e) and scale (press e) in front view. Give it a colour. In the edit buttons menu press "Set Smooth". Then copy it (shift + d) and move (press g) it to

the other side.

In side view copy (shift +d) the squeezed sphere, scale it down (press s). Then in front view move it out of the big sphere to make it visible. Copy it (shift +d) and move (press g) it to the other side.

The lid

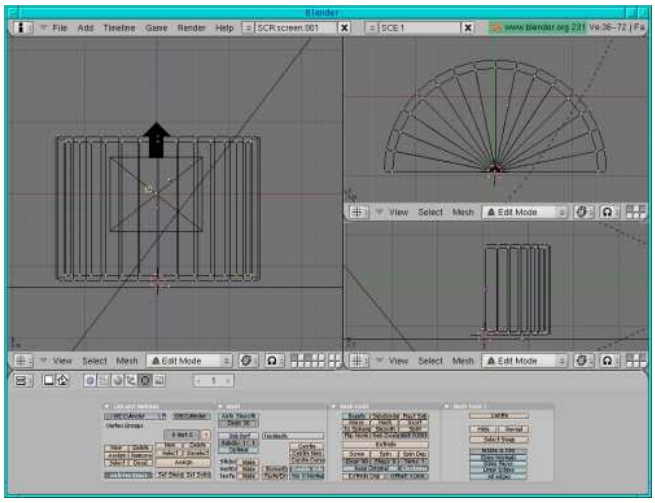
For the lid add a cylinder (hit Space, -->Add-->Mesh-->Cylinder, let the vertices have a value of 32) in top view (in the third layer). Press a to deselect all points, then press b and mark the bottom half.



Next press x and delete the vertices.

Press a to select all points, then still in top view press e (to extrude), enter and s to scale everything a bit down.

Now press g and move the inner half-cylinder a bit down so that its bottom line is exactly on the bottom line of the outer half-cylinder.



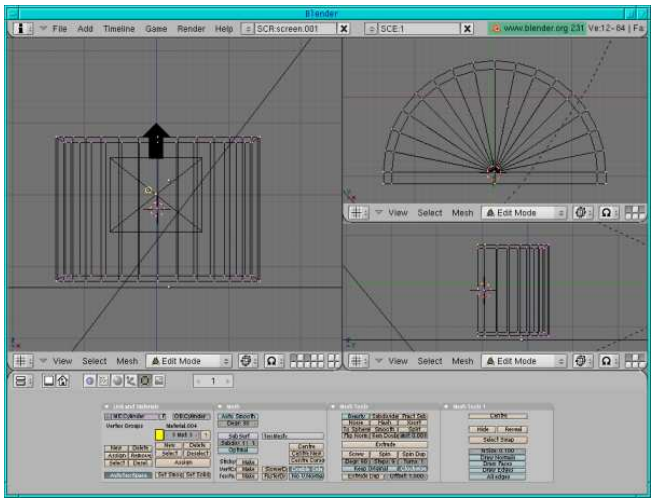
Leave the inner half-cylinder still selected. Give the "two" half-cylinders a pink colour (go to the material/shading buttons, press "add new" and move the colour sliders to R=1, G=0, B=1). Now go to the edit buttons and assign a new colour to the inside of the lid as it should be of a lighter pink colour: Press "New", "Select", then go to the material buttons and assign the new light pink colour with "Add new", then colour sliders to R=1, G=0.6, B=1, then back to the edit buttons and press "Assign". Now make a render (F12) to see if everything is correct. A simple model of our lid is ready now.

The upper edge of the lid should be yellow. So in top view select the bottom line (press a to deselect all points, then b and mark the line) and assign a yellow colour: like before go to the editing buttons, press "New", "select", go to the material buttons, press "Add new" and move the colour sliders to R=1, G=1, B=0, then go back to the editing buttons and press "assign".

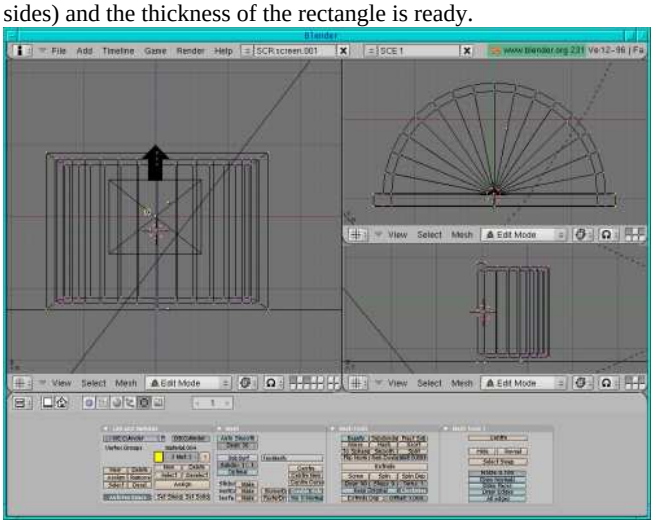
Now for the thickness of the rectangle of the lid:

Select the bottom line in top view (press b and mark the line) if it isn't still selected.

In top view press e, enter, arrow key to move the points a bit down, enter. Still in top view select the two outer points on the two sides plus all inner points in the middle of the rectangle, then in front view deselect the two inner points in the middle (see figure below).

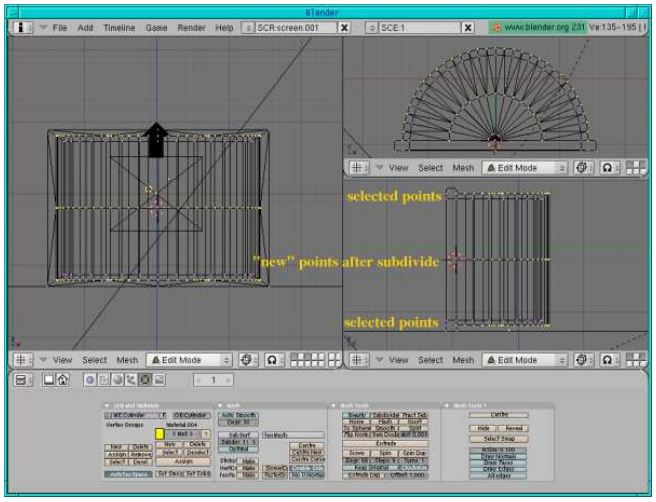


In top view press e ,enter, s, arrow key, enter (to scale up to the sides) and the thickness of the rectangle is ready.



Now for the thickness of the arcs:

In side view select the outer arc on bottom and top, then press "subdivide" (you find the button in the edit buttons under "Mesh tools", next to "Beauty" and "Fractal Subdivide").



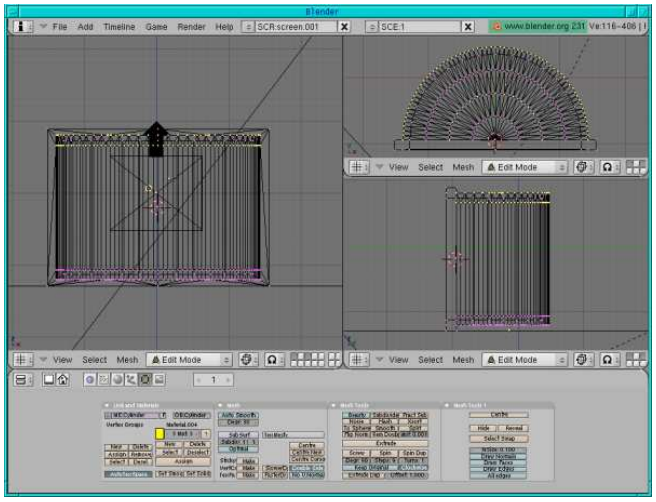
As we need the thickness on both sides we need to s

ubdivide again: in side view deselect (press b and mark with the mouse, then right mouse click) the top line of the arc and press subdivide again.

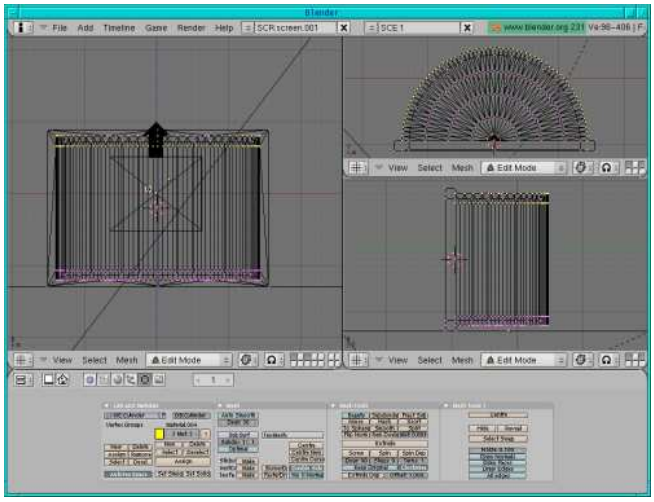
In side view deselect the top and bottom lines that are selected, press g and arrow key to move the arc line downwards, then press enter to finish the operation.

Now select the arc points that we obtained through our first subdivision, press g and arrow key to move this arc line upwards, then enter.

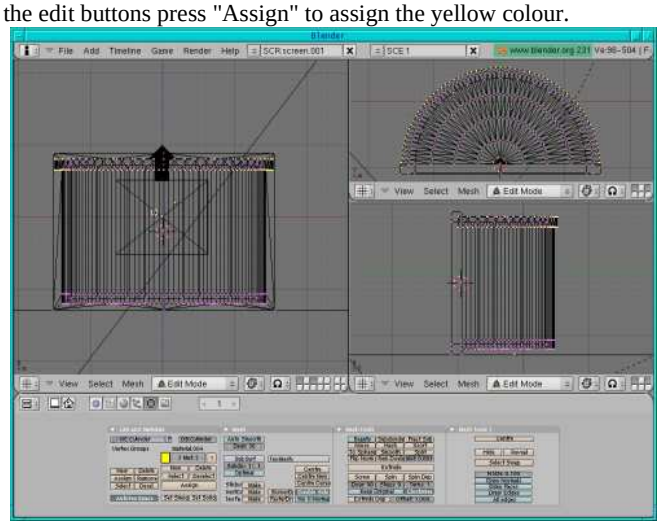
In side view on top select the first and third line (see figure below, don't select the point that belongs to the rectangle).



In top view deselect the inner circle and inner points.

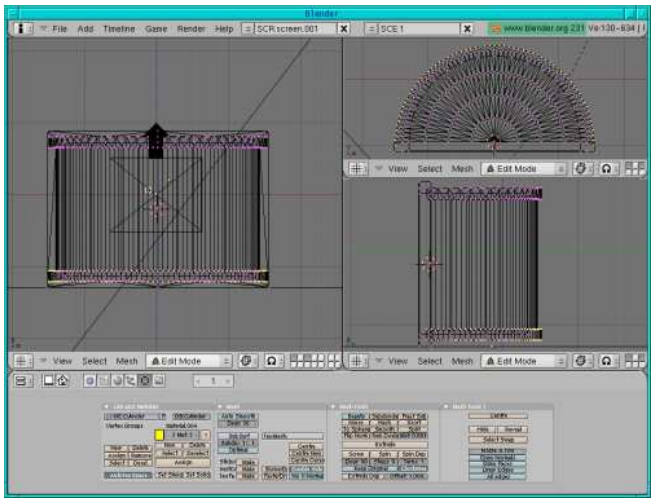


In top view press e,enter, s, arrow key (to scale the arc up), enter. In the edit buttons press "Assign" to assign the yellow colour.



Now the other side:

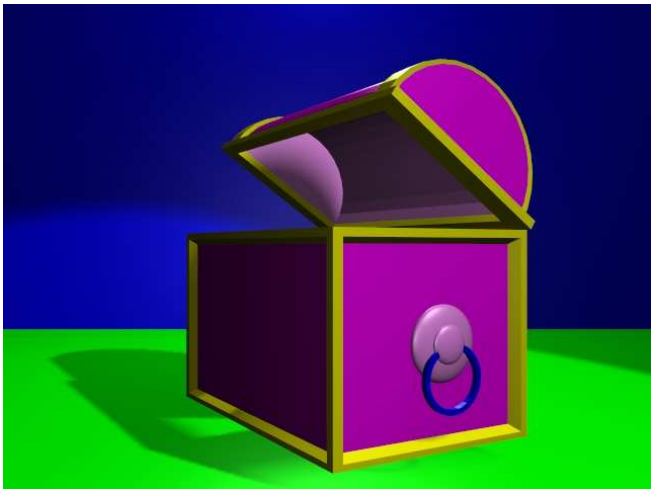
In side view select the first and third line on the bottom. In top view deselect the inner circles and inner points. Still in top view press e, enter, s, arrow key (to scale the arc up) and enter when you have exactly reached the other arc points. In the edit buttons press "Assign" to assign the yellow colour.

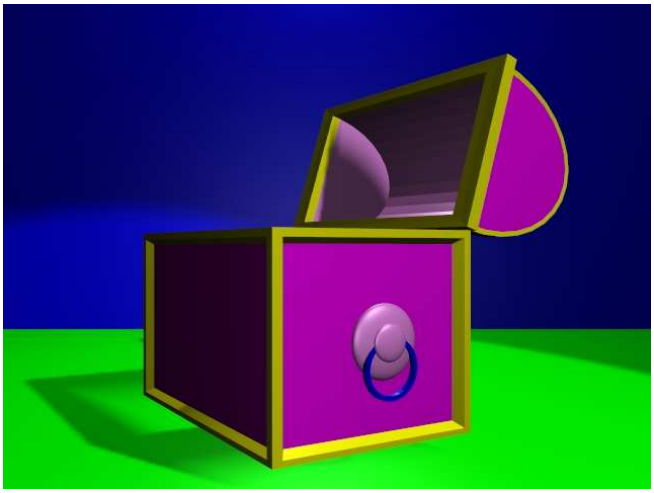


The lid is ready!

Putting box and lid together

The lid is in layer 3 and the box in layer 2. Make both layers visible, turn the lid 90 degrees, then move the two over each other and scale them. That's it.





Have fun and happy blending! :)

References

- The Official Blender site (here you get the latest information about the further development of Blender, you can download it, there are tutorials ..): http://www.blender.org
- Blender cafe (in English and French): http://www.linuxgraphic.org/section3d/blender/pages/index-ang.html
- Elysiun site: a Blender community site:http://www.elysiun.com
- General articles about 3D graphics and animation: http://webreference.com/3d/

...

When complete, mail back to the editor at <auugn@auug.org.au>

This article is re-printed with permission. The originals can be found at:
http://www.linuxfocus.org/English/January2004/article325.shtml

Tuxpaint: A paint program for kids

Author: Katja Socher <katja@linuxfocus.org>

Abstract

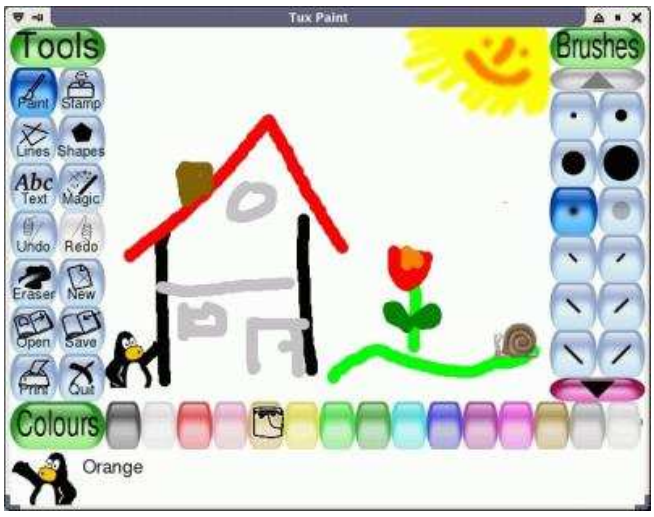
Tuxpaint is a paint program (not only) for children that is absolutely great and fun!

Introduction

A few weeks ago I was looking for software for kids in the internet and so by chance I found Tuxpaint which looked interesting. When you install it you can choose between several languages so that there is also a big chance that your and your kid' mother tongue is among the available languages which is important for software for children. The installation went quick and without problems. So soon I was ready to play around and once started I found I could hardly stop again. :)
Even though it is rather simple when you compare it with graphical programs like The Gimp it is full of interesting and easy to grasp features!

Simply click on the Paint button and start drawing. You can choose between many different brushes and colours. If you want to draw a line you press the lines button and a click on the shapes button lets you choose between many different shapes. You can even rotate and scale your shapes before they appear on your drawing board.

For text you click the ABC text button, choose your colour and font and here you go. The text appears in a box and can be moved with the mouse until you confirm it with "enter". "Print" let's you print out the picture.



In case you didn't like what you painted you have several undo steps and redoing something also is possible. You can erase part or all of your drawing or if you want to start with a clean sheet you simply press the "new" button.

To save your picture click the "save" button and to open an existing picture you click "open" and Tuxpaint shows you thumbnail images of your saved pictures.

The pictures are saved as png files in the directory .tuxpaint/saved so you can copy them to another directory before your daughter or son changes her/his great painting of an elephant into dracula again.

All this already sounds great, doesn't it? But the real cool features are the stamp and the magic buttons!
A click on the magic button and you can paint in rainbow colours, you can add sparkles to your pictures, give your picture a special note with the chalk option, you can blur your picture and more!
Stamps are what in other drawing programs is called cliparts and there really are a lot of images! Changing the colour let them appear in this new colour and you can also change the size as well as flip them! I really was impressed by the variety of stamps that come with Tuxpaint!

To add your own stamps with (or without) sound you need a small png file and add it in one of the subdirectories under whereyourtuxpaintisinstalled/share/tuxpaint/stamps/, e.g. /usr/local/share/tuxpaint/stamps/misc/symbols/shapes/mypicture.png
Optionally you can save /usr/local/share/tuxpaint/stamps/misc/symbols/shapes/mypicture.txt to get some text displayed, e.g. This is my picture stamp. And if you want sound for your stamp you also need a wav. file: /usr/local/share/tuxpaint/stamps/misc/symbols/shapes/mypicture.wav.

Most button clicks are accompanied by sound that make the painting even more fun! So far my favourites are the sound of the rainbow button and the car racing sound of the checkered flag. But decide for yourself!

I think that kids would love this program. At least I love it as painting with it really is such a big fun! And I can not only just mess around but even create nice looking greeting or invitation cards!

It's time now to let you go so that you can explore Tuxpaint on your own!

Thanks to the Newbreeds Software people who developed this wonderful tool!!!
And to the rest of you I just want to say:
Happy Tuxpainting! :)

References

You can find and download Tuxpaint at
http://www.newbreedsoftware.com/tuxpaint/

From there you can also get the "Rubber Stamps Collection".
...

When complete, mail back to the editor at <auugn@auug.org.au>

This article is re-printed with permission. The originals can be found at:
http://www.linuxfocus.org/English/March2004/article333.shtml

AUUGN CD-Rs in this issue

The bad news first. We don't have a CD for you in this issue.

The good news. We're prepping a four CD set which will be sent to AUUG members separately.

Who Are You?

The AUUG'2004

Annual Conference

Melbourne, 1-3 September 2004

Tutorials 29-31 August 2004

Call for Papers

As more devices, companies and people get connected to the Internet, computer security becomes increasingly important. And often security boils down to three things:

- **Identification** – working out who you are dealing with.
- **Authentication** – confirming you know who you are dealing with.
- **Authorisation** – letting the known person do what they are allowed to do and no more.

With that in mind, AUUG has chosen as the theme for the 2004 conference: "Who Are You? Identification and Authorisation Issues in Computing.", and invites proposals for papers and tutorials relating to:

- Identification, authentication and authorisation
- Applications of cyptography and cryptographic protocols
- Maintaining privacy
- Achieving anonimity on the Internet
- Internet security
- Other aspects of computer security

We also call for papers relating to topics of general interest to AUUG members:

- Standards based computing
- Open source projects
- Business cases for open source
- Open source in government
- Technical aspects of Unix, Linux or BSD
- Performance measurement and management
- Software development
- Networking, Internet and the World Wide Web.

Presentations may be given as tutorials, technical papers, or management studies. Technical papers are designed for those who need in-depth knowledge, whereas management studies present case studies of real-life experiences in the conference's fields of interest.

A written paper, for inclusion in the conference proceedings, must accompany all presentations.
Speakers may select one of two presentation formats:
Technical presentation: a 30-minute talk, with 10 minutes for questions.

Management presentation: a 25-30 minute talk, with 10-15 minutes for questions (i.e. a total 40 minutes).

Panel sessions will also be timetabled in the conference and speakers should indicate their willingness to participate, and may like to suggest panel topics.

Tutorials (held 29-31 August) provide a more thorough presentation, of either a half-day or full-day duration. They may be of either a technical or management orientation.

The AUUG'2004 conference offers an unparalleled opportunity to present your ideas and experiences to an audience with a major influence on the direction of computing in Australia.

Submission Guidelines

If you are interested in submitting a paper you should send an extended abstract (1-3 pages) and a brief biography, and clearly indicate their preferred presentation format.

If submitting a tutorial proposal you should send an outline of the tutorial and a brief biography, and clearly indicate whether the tutorial is of half-day or full-day duration.

Speaker Incentives

Presenters of papers receive free registration to the conference (1-3 September), including social functions, but excluding tutorials.

Tutorial presenters may select 25% of the profit of their session OR free conference registration. Past experience suggests that a successful tutorial session generate a reasonable return to the presenter.

Please note that in accordance with GST tax legislation, we will require the presentation of a tax invoice containing an ABN for your payment, or an appropriate exempting government form. If neither is provided then tax will have to be withheld from your payment.

Important Dates

Abstracts/Proposals Due:	7 May 2004
Authors notified:	4 June 2004
Final copy due:	2 July 2004
Tutorials:	29 to 31 August 2004
Conference:	1 to 3 September 2004

Proposals should be sent to:

AUUG Inc.
PO Box 7071
Baulkham Hills BC NSW 2153
Australia
Email: auug2004prog@auug.org.au
Phone: 1800 625 655 or +61 2 8824 9511
Fax: +61 2 8824 9522

AUUG 2004 Information

For general information on the conference, including discussion of sponsorship, advertising and display opportunities, please write to: <busmgr@auug.org.au.>

Details on the conference programme, including queries regarding paper submissions, or ideas for speakers, papers or tutorials you would like to see at the conference, should be sent to: <auug2004prog@auug.org.au.>

Alternately, all queries can be handled by the AUUG Business Manager, Liz Carroll. Liz can be contacted by telephone on 1800 625 655 or +61 2 8824 9511, by email to <busmgr@auug.org.au>, or by facsimile on +61 2 8824 9522."

Please refer to the AUUG website for further information and up-to-date details:
http://www.auug.org.au/events/2004/auug2004/

2003 And Beyond: Final

Author: Andrew Grygusr aax@aaxnet.com

[Editorr's note: This is the final part of this serious We will include the references to all the previous articles at the end of this one.]

MICROSOFT'S LEGAL PROBLEMS

Microsoft's endless legal problems will continue to erode the company's public image and the trust of business partners.

Even the antitrust case, which Microsoft has settled with the Department of Justice, continues to grind on. Microsoft executives have freely admitted the settlement they negotiated with the Bush/Ashcroft administration hands them **greater power than they had** before the trial began - not surprising **since they wrote it** and Bush/Ashcroft just retyped and signed it, but others continue to pursue the case,

Microsoft's highly publicized moves to "comply with the DoJ settlement" are PR stunts and are causing adverse publicity. Their high profile release of 272 APIs (Application Programming Interfaces), for example, was in a format completely useless to anyone (Q1). In July 2003, the DoJ have reported serious concers to the Court which may result in further action (as they have done with previous DoJ agreements, judge CKK can pretty much have her way and her way of having it as far as remedies are concerned.

Microsoft's program to license other APIs involves unacceptable terms and unacceptable costs, so only four long time Microsoft partners have signed up (Q15). The DoJ is cites concern, but will try to get Microsoft to ease the terms before taking the matter back to Judge CKK.

Questions have now surfaced about claims Microsoft made during the antitrust settlement hearings. Microsoft argued that exposing the Windows source code would seriously compromise national security. Now they have agreed to open this same source code to the governments of Russia and China (X62). **Is Microsoft deliberately compromising American national security** to protect its markets, or have they been bending the truth in court (X63)? Either way, they should be held accountable.

Microsoft also told the judge that making a modular version of Windows as desired by the States was all but impossible and would destroy the product, forcing them to withdraw Windows from the market. Now, starting with Windows 2003 Server and particularly future products, Microsoft's primary development thrust is to make their products modular. Did they lie to the judge?

There are currently about **25 patent infringement cases** against Microsoft (Q4). Technology innovators tend to be hopelessly naive and do not recognize the smell of Microsoft's money as the smell of death. Microsoft routinely drags out license negotiations until they know everything they need to know about the product. Negotiations are suddenly dropped, and Microsoft issues an infringing product. The next step is to cut off the innovator's cash flow and use Microsoft's massive legal forces to bankrupt him in court. No royalties will be paid.

The list keeps growing. British company **Sendo** partnered with Microsoft to bring Microsoft's SmartPhone version of Windows to the mobile phone market. Just days before introducing their product, Sendo dropped the relationship and has filed suit against Microsoft. It seems Microsoft's favored contract manufacturers in Asia, who have little software design expertise, started shipping handsets incorporating sophisticated technology **identical to that developed by Sendo (Q6).**

Microsoft has just lost its appeal of an important patent case brought by Timeline. Microsoft signed a patent license agreement with **Timeline**, but apparently **issued misleading statements** about that agreement to their customers to encourage developing for SQL Server. SQL Server developers now face the possibility of having to pay staggering license fees to Timeline. Some are considering suing Microsoft for misleading them .

Two cases cleared for trial **threaten every version** of Windows from Windows95 on, and if either wins, it will cost Microsoft the big bucks. Microsoft does lose in court pretty regularly, since they're clearly guilty most of the time. When they lose, they negotiate a settlement incorporating a **nondisclosure agreement**. This costs them extra, but is well worth the cost, because the public never learns the true extent of Microsoft's violation of laws and ethics.

In fact, nearly every Microsoft agreement or contract carries an NDA (Non Disclosure Agreement), often to hide the extent to which extortion was applied. Facts hidden by the NDA may seep out many years later. The NDA required to get critical Microsoft programming assistance in the transition to Windows, for instance, included an extortion clause requiring software developers to **drop all development** of software for IBM's **OS/2** operating system. By such means, superior products are driven from the market.

Now, with Microsoft battling Linux in the universities, they require NDAs that force public institutions to be **in violation of U.S. public records laws**. In other words, "You want our software, you're going to have to break the law a little".

Related legal problems are that Microsoft's licenses for Windows XP and Windows 2000 SP3, which are expected to be expanded in future products, apparently force customers to **violate both U.S. Banking law and HIPAA (Health Insurance Portability and Accountability Act of 1996**. Microsoft's demand to be allowed to enter, examine and make changes to systems that are required to be certified and secured is incompatible with U.S. law and regulations.

Microsoft also faces **legal problems oversea**. Most prominent is action by the European Commission which is investigating Microsoft's business practices in several areas. If charges are brought, the EU has much less incentive to accept a favorable settlement than Bush/Ashcroft did. The **Taiwan FTC** (Fair Trade Commission) also has ongoing action concerning Microsoft's business practices.

For certain, there will be many more legal actions against Microsoft, because the company has never respected the law. Microsoft started out by stealing intellectual property ("dumpster diving" for code) and computer time, and it hasn't changed its attitude one bit since. It is likely **new Federal antitrust charges** will be filed soon after Bush/Ashcroft leaves office - there is certainly plenty of material for one.

The smiling Bill Gates mug shot from 1977 exemplifies this attitude. Microsoft claims it was for a traffic violation, but the **laws of New Mexico say otherwise** - mug shots are reserved for more serious offenses. What was the offense? Microsoft's money has

caused that to vanish from the record. As long as money buys "justice" Microsoft will respect neither law nor justice.

WHY CHOOSE MICROSOFT SOLUTIONS?

Despite matters discussed above, some of which can be interpreted as detrimental to businesses, **most businesses Will choose Microsoft solutions**, and many will choose **only Microsoft solutions**.

A principal factor is that America's business leaders simply don't want to think about complex technology issues - **they want to think about golf**. Microsoft promises them that, and being a large, and hugely successful corporation, they have high credibility with top business executives.

Microsoft's sales teams waste little time pitching to people who understand and implement technology, they pitch to executive "decision makers" who have the **power to dictate** what information systems will be used, even though they know little about them. Microsoft has direct access to high level managers, many of whom are **strong admirers of Bill Gates' wealth,**

So strong is the desire to simplify decision making, many executivs are willing to live with "solutions" that don't actually work. The industry is replete with stories of "Microsoft only" shops where staff replaced Microsoft products that didn't work well with lower cost products that did, only to be ordered to remove them and return the Microsoft "solution" to service. Other shops are reported to have adopted a "don't ask, don't tell" attitude.

Microsoft promises integrated solution packages they assure the customer will **all work together seamlessly** to implement **efficient new business processes** resulting in **huge contributions to profitability** in an **amazingly short time**. Better yet, this is all pre-packaged and can be implemented and run by cheap, semi skilled labor rather than the expensive administrators required by other systems - it's all "point and click".

Microsoft backs all this up with very well developed product selection tools (M2) to help configure systems that might actually work. If the tools are good enough that semi-skilled managers feel comfortable pointing and clicking their way through the design stage, those expensive and troublesome "experts" won't be brought in at all, and **superior alternatives** to Microsoft's solutions **will not be examined**.

That's the pitch: at the top, become a superhero to the stockholders without taking your mind off golf, and for the middle managers, the security that "Nobody ever got fired for buying Microsoft" (not actually true, but widely believed). This two-pronged pitch is difficult for any competitor to counter.

Small businesses don't get wined, dined and golfed by Microsoft sales, but they are still continuously exposed to Microsoft's marketing materials, and are just as vulnerable to the temptation of making easy choices. "Lets just go with the leader - that'll be safe." In some cases small businesses have little choice, since their internal systems are dictated by customers and business partners much larger than they are, and they don't have the skills to get around that.

Most small businesses don't call in a consultant until they have **already decided on a course of action** - they just expect the consultant to make what they have decided to do work. In truth, most consultants are happy to go along with this, because they can charge **full rate for everything**, and not get blamed for the choices. A system that "sort of works" **generates a lot more consultant dollars** than one that works (IBM Global Services favors Windows over IBM's own OS/2, because OS/2 works).

Does Microsoft deliver on its promises? In many ways, yes - but that "bottom line" part - no. Microsoft's solutions have always proven to be **very expensive**, sometimes absurdly expensive, and often far more expensive than available alternatives. If Microsoft solutions were cheap, Microsoft wouldn't be that rich.

Microsoft's solutions often look inexpensive, and are always claimed to be, but are often cheap by the unit and expensive overall. There is often little economy of scale. One unit, a thousand units, the cost per unit is about the same, and each carries full support costs. There are also often many more units than with competing solutions, especially when it comes to servers.

Some years ago, Microsoft convinced management that moving from Novell NetWare servers to Microsoft Windows NT servers would **save them huge amounts of money**, because the servers could be administered by people with far less skill - it's all "point and click". Companies following this path found that every NetWare server was replaced not by one, but by **three or four** NT servers, and the admins were definitely cheaper, but there were f**our or five times** as many of them.

Even Microsoft internal documents admit that, dsspite Windows' "point and click" interface, Windows server administration is more difficult and time consuming than with Unix/Linux (A15). Others have had similar results (C37). Compound this with constant security patches and a platform that crashes much more often and you start to see why you need more administrators.

Why wasn't a big stink raised? For the pointy haired bosses of middle management, the **Windows** conversion was a godsend. Four or five times as many employees means **more power and more pay** - and nobody could object or criticise, because it was a**ll dictated by top management**. These same conditions apply on the "Road Ahead".

This state of affairs **will persist**, and be resistant to change. Business leaders simply are too comfortable with the Microsoft solution, and become more comfortable as Microsoft eliminates more competitors - **less decisions to be made**. Being able to say, "we really had no choice", is something most business "decision makers" are willing to pay any price for.

Alas, this soothing ointment does come with flies in it. There are many applications for which Microsoft solutions are simply **not at all suitable**, and others where they become **way too costly** the moment a competitor implements an alternative solution. This has given alternatives staying power, and as the cost and complexity of Microsoft's solutions continue to increase, alternatives increasingly threaten the status quo.

MICROSOFT'S COMPETITORS

Given that Microsoft makes such a compelling case to business leaders, you would think competing products are at the end of their days, yet many are showing such strength, even the technology

press is beginning to notice.

In important areas, competitors have slowed Microsoft's expansion to a crawl, and even threaten to recover lost territory. Deliberately incompatible Windows features now often slow acceptance of Microsoft products rather than demolish competitors as intended. To maintain revenue growth, Microsoft has been forced to increase costs to its current customers, providing yet more incentive to look at alternatives.

What happened? Windows NT was supposed to hit Unix hard (it did - like a bug hitting a windshield), and the last mainframe was to be unplugged before the end of the century - yet Unix and mainframes are still the power houses of business technology. Even Novell, decimated by Microsoft's superior marketing, is showing signs of renewed life.

The problem is simple, superior marketing can take you a long ways, gathering all the low hanging fruit, but eventually you get to a place where you **must compete on merit**. Microsoft has reached that place, and their products have proven inadequate for many jobs.

Business with highly demanding applications requiring outstanding performance, vast storage capacity, and high reliability find Microsoft products do not meet cost / performance and manageability requirements. Embarrassingly, Microsoft's own HotMail service falls within this category. Unix, Linux and mainframes each have established markets here.

Specialized workstations - Many organizations have a large number of workstations that don't need Microsoft Office functionality, they just perform one or a very few specialized functions. These stations can be rolled out on thin client systems or Linux workstations at far lower cost, especially for ongoing administration (no - Windows Terminal Services is not equivalent).

Supercomputer class performance is needed for oil exploration, movie special effects, weapons research, weather analysis and many other demanding applications. Microsoft has no product at all for this market - Linux clusters now all but own this space and even Microsoft's own researchers have endorsed Linux clusters for supercomputing applications.

Cost - Fast growing new businesses with limited financial resources are turning to alternatives, particularly Linux. In years past, cash limited businesses just copied a lot of Microsoft software, but this is becoming difficult and dangerous. Many businesses in highly competitive financial services, volume retail and fast food are running alternatives to Windows.

Business Continuation - Some businesses have taken a hard look at issues of data ownership, control, security, and business continuity (see above), and realized their interests and those of Microsoft are on a collision course. Yes, there are a few business executives who don't play golf.

Ethics - Organizations that put a premium on ethics (few businesses, but some social organizations) find it impossible to rationalize using Windows. Only our top corporate CEOs (such as they are) could consider Microsoft to be an ethical company.

Server Consolidation - Windows NT/2000 server is slow, runs only on relatively weak Intel based platforms, and major Windows server applications are not at all happy to share the same server. On converting to Windows, many companies found they had four or five Windows servers where there had been only one NetWare server, and more were added as needs expanded. With Windows being so high maintenance, this resulted in a huge increase in cost, complexity and staffing. To reverse this, many companies are adding multiple Linux partitions to their IBM zSeries and iSeries servers to get their operations back on a sane number of servers.

License Raids - Some who have suffered a BSA / Microsoft license raid have stripped all Microsoft software from their business to make sure it never happens again. A license raid feels like rape, but costs a lot more. A few businesses believe in prevention, and are moving to alternatives, particularly Linux, before getting raided. All you need to get raided is one disgruntled employee (whose identity will be kept secret.

Specialized Devices - Makers of consumer electronics and other specialized devices have turned away from Windows on grounds of cost and flexibility. Despite investing billions in cable companies, Microsoft has lost almost the entire interactive TV market to mostly Linux based alternatives, and Symbian dominates in advanced cell phones (and Linux coming soon to a Motorola phone near you).

Security - Some businesses are actually bothered by Microsoft mucking around in their data systems on a regular basis, and that crackers and spies find easy pickings. They may even resent every worm, virus, and trojan having its way and its way of having it with their PCs. Windows security - it is to laugh! There are over 63,000 worms and viruses for Windows, many out of control, under 100 for Linux, and none that amount to much.

WHAT ALTERNATIVES ARE THERE?

Linux - Desktop & Server - Now Number One on Microsoft's enemies list, Linux is eating server markets Microsoft expected to be theirs, and increasingly threatens their desktop monopoly. Linux is an updated, more user oriented version of Unix. It scales from wristwatch to supercomputer. We have a lot more to say about Linux below.
Disadvantage: many specialized business applications don't run on Linux, yet (but many do and more are coming).

eComStation (OS/2) - Desktop & Server - IBM's OS/2, offering the most usable desktop environment on PCs, is now updated, enhanced and distributed by Serenity Systems as eComStation (by contract with IBM). It's economical, secure, free of worms, virus, trojans, crackers and license raids. It's easy to use, and plays well with other systems (Windows, Linux, etc.). It can be outfitted to run Microsoft Office for light usage.

At Automation Access, we run our business on OS/2 (including building this Web site) and have no intention of changing - anything else would be less stable and cost a lot more. eCS / OS/2 also makes a fine client to Linux or DOS based accounting systems.

Most OS/2 users are larger organizations that depend on critical information systems (banks, insurance companies, airlines, grocery chains, etc.).
Disadvantages: IBM wishes it would go away because something that doesn't break doesn't generate service revenues (why IBM Global Services loves Windows). Support for the latest cameras, scanners, etc. may be slow coming out, so most OS/2 based offices keep a Windows machine around for that.

Thin Clients / Java - Desktop/Server - in many organizations, there

are a lot of workstations where a full Windows desktop is not justifiable, where Thin Client is fully adequate and can save big bucks in support and maintenance costs (all the software is in one place, on a Linux, Sun or IBM server). If a Thin Client fails, you unplug it and plug in another - that's all.
Thin Clients generally have no hard disks and no programs beyond what's needed to find a server to boot from. They may boot a stripped down Linux with X Windows, a minimal Java OS, or some other simple system with a Web browser. The coming era of Web Services will result in massive growth for thin clients.
Disadvantage: they don't run Microsoft Office (but do run StarOffice / OpenOffice).

Microsoft's WTS (Windows Terminal Services) will run Microsoft Office, but it's hardly "Thin Client", it's more like "Fat Server". If you have to have Office on Thin Clients, Web based solutions are provided by Citrix and Tarantella.

Apple Macintosh - Desktop & Server - Heavily used in publishing and advertising, the Mac also sees use as a general purpose small business system (A5). Apple's OS X (an Apple user interface running on a BSD Unix operating system) has caused renewed interest in Apple computers. Microsoft Office is available for OS X.
Disadvantages: you are tied to Apple's hardware, and the selection of business software is relatively small (though growing, especially since Linux software is easily ported to OS X).

Novell NetWare - Server - Often used to support networks of Windows workstations to provide greater security, better performance and lower costs than with Microsoft servers.
Disadvantages: no desktop environment, and a shrinking pool of techs who understand NetWare administration.

VMS - Host, Server - DEC's (Digital Equipment Corp) VMS dominated the minicomputer field when minicomputers dominated, VMS is still considered by many to be the "One True Operating System", and it is still widely used.
Disadvantages: DEC was bought and dismantled by Compaq, which preferred to sell Windows. Compaq was bought and dismantled by HP, which would rather sell Windows and Linux.

Unix (Commercial) - Host, Server & Engineering Desktop - Whether supporting "green screen" terminals, thin clients, or Networked PCs, Unix is the workhorse of the server room, and runs on Sun, Intel, IBM and many other platforms, Unix is also the platform for thousands of specialized "vertical market" software packages. It runs the McDonalds restaurant chain, your local telephone switching system and most of the entire Internet, as well as many small business accounting systems. After many years of trying, Microsoft is unable to move its HotMail service from Unix to Windows.
Disadvantage: why run Unix when Linux is a more modern version, costs a lot less and is easier to support?

BSD Unix - Server, Host & Development Workstation - BSD Unix is most used by ISPs (Internet Service Providers) and by software developers. The several varieties of BSD Unix each serve a different audience and each has a separate development group with unique goals. Several versions of BSD Unix are free and open source.
Disadvantages: Little known to the general business community - otherwise same disadvantages as Linux

QNX - Desktop, Industrial controls, Automobiles - QNX is a Unix like operating system used for applications requiring fast "real time" response and which absolutely must not fail, ever. It is highly modular so no unneeded components have to be installed. IBM has selected QNX for its automobile navigation system. Microsoft likes to think Windows CE competes with QNX, but that's hardly the case.
Disadvantages: not for the general purpose desktop.

IBM iSeries (AS/400) - Host/Server - scaling from small business to major enterprise, iSeries is for businesses that require the highest level of reliability and solid performance. For many years, Microsoft's deepest darkest secret was that their business management and accounting ran on AS/400, not Windows. iSeries is now much more flexible since it can run Linux alongside its regular tasks (or Linux only) for server consolidation and expanded application availability.
Disadvantages: few small business people are aware of it.

IBM zSeries (mainframe) - Host/Server - for system that must support thousands of users, thousands of transactions a second, Terabytes of data storage and NO downtime. Mainframes are now more flexible, since they can run even thousands of instances of Linux alongside their regular tasks (a "Linux only" version is also available).
Disadvantage: you've got to have some really, really serious transaction and storage demands to justify the cost - this is not a small business platform.

Supercomputers - Compute Engine - Once the province of highly specialized computers costing millions, most supercomputers are now large clusters of low cost computers running Linux. A few traditional supers are still made for applications requiring truly linear processing.
Disadvantage: you have to need a really, really serious compute engine to justify the cost.

The prominence of **Linux** in every one of our "reasons" categories, and its appearance in many of the "alternatives" as well, has caused Microsoft's management to move it to the top of the "Enemies List", and declare it to be the most serious challenge Microsoft has ever faced.

To date, Linux has displaced Unix to a greater extent than it has Windows, but that is changing. A new report from Forester Research states that Windows servers are increasingly replaced by Linux (C36). Even where Linux replaces Unix, it's a loss for Microsoft, because they expected all those Unix servers to be replaced by Windows.

Linux gained a foothold in server conversions because it's much easier to convert from Unix to Linux than from Unix to Windows. Once Linux is in place, Windows no longer offers a cost benefit. With Linux deployed, many companies found they preferred it's stability and ease of administration to Windows servers. This has seriously stalled Microsoft's expansion in the server market.

To make matters worse, developers are rapidly improving Linux' already capable **graphic desktop environment**, which can even be configured to look exactly like Microsoft Windows. As Linux starts seeping out of the server room into "line of business" workstations, Microsoft's most jealously guarded monopolies are directly impacted.

If you don't think American business is taking Linux and other open source products seriously, you might check out a recent article in

CIO (Chief Information Officer) Magazine, aimed squarely at corporate officers, and a Computerworld article in an issue aimed at helping corporations falling behind the technology curve catch up.

Microsoft's crown jewel has always been control of software developers, enabling them to starve other platforms for software titles, but a recent study by Evans Data, a research group serving the developer market, has found developers abandoning Windows for the Linux platform in unexpected numbers. This is perhaps the worst news yet for Microsoft's future.

Linux is very difficult for Microsoft to fight, because it isn't the product of a single company they can buy out or bankrupt. The Linux code is open source (free and freely available), so anyone who wants to can publish it, and thousands of programmers worldwide contribute to its improvement and maintenance. Destroy one Linux publisher and another would take its place overnight.

Evidence indicates Microsoft's most successful tactic is pressuring manufacturers to supress Linux on their equipment by not providing drivers or marketing support. For instance, Intel, which has always enthusiastically supported Linux in the server market (which Microsoft does not control), withdraws support entirely if the desktop (which Microsoft does control) is involved. While this has antitrust implications, what's actually happening is, as always, concealed by Microsoft's insistence on NDAs (Non Disclosure Agreements).

Linux isn't the extent of the problem, though. Other open source products counter Microsoft, often in conjunction with Linux, but often on Windows itself. open source **Apache** dominates the Web server market with a 60% share, while Microsoft's IIS holds less than half that. Additionally, **IBM's Websphere** e-commerce suite is also based on Apache. open source **OpenOffice** and its commercial variant, **StarOffice 6**, now directly threaten the Microsoft Office monopoly. When our clients balk at $500/workstation for MS Office, we just install OpenOffice for free.

With business management moving to Web Services and Web based applications, the strength of open source Web applications is very worrisome for Microsoft. For instance, both **FedEx Freight** and **Union Pacific Railroad** have placed their customer interface and traffic management systems on Linux and/or Apache. Neither FedEx Freight nor Union Pacific Railroad are exactly "Mom & Pop" operations.

FedEx Freight moved from Windows NT to Linux / Apache, and intends to dump another 40 or 50 Windows servers, consolidating their functions onto a single 4-processor Intel server running Linux. Union Pacific has a mostly Unix network, and refers to their remaining Windows / IIS applications as "legacy applications".

So powerful is the Linux freight train, even mainline business magazines are getting on board. Business Week has just published a 9 article cover feature on Linux.

Cost and Competitiveness Issues

Competitiveness issues are arising from the high and rapidly rising cost of Microsoft solutions. Some years ago, research firms Forester, Gartner and others established the real cost of Windows PCs on a corporate network to be between **$8000 and $14,000 per year per PC**, and corporations confirmed these figures internally (the bulk of this cost is hidden, consisting of support, administration and upgrades). Multiply this by the number of PCs in the company and it starts to look like real money.

As long as your competitors use similarly costly systems, you can just pass costs on to customers with little loss of business, but once a competitor implements a more cost effective system - you have a problem.

This is now happening in the financial markets of Wall Street. **Morgan Stanley**, **Merrill Lynch** and others have moved their financial reporting systems to Linux, and now everyone else is scrambling to follow suit to bring their costs in line. Some Windows NT/2000 is being replaced, but much of the transition is from Unix. Nonetheless, it's a major loss to Microsoft because they expected to own this territory, and now it's gone.

The same thing happened in the special effects industry. **Titanic** was rendered on a Linux supercomputer cluster. **Lord of the Rings** was produced on Linux workstations and rendered on Linux supercomputers. Now the entire industry is either on Linux or scrambling to get there as fast as possible, including **Disney** and **Lucas**.

Retail is already starting to slide down the slippery Linux slope, with **Sherwin Williams**, **Papa John's Pizza, Burlington Coat Factory, Boscov Department Stores, Regal Entertainment Group** (550 theater concession stands), **Hannaford Brothers** (119 grocery stores) and others. It won't be long before this becomes the same sort of landslide the special effects business has experienced. Sending millions upon millions to Microsoft is hard to justify in hardscrabble retail, especially if your competitor isn't.

Other businesses are looking very hard at Linux. Even **GiftCertificates.com**, founded by a former Microsoft executive, and built entirely on Microsoft software, is now seriously considering a move to Linux. The difference "is hundreds of dollars on the Linux side versus tens of thousands of dollars for Windows." according to its CEO (C14).

Even the **U.S. military** is concerned with costs these days. A report prepared by MITRE for the U.S. Army is very favorable toward the use of Linux and open source software for government systems, even though Microsoft was allowed to review the report and request changes before it was published.

Linux appears in every category Microsoft operates in and beyond in both directions (wristwatches to supercomputers). It cooperates well with other threats such as Java, Thin Clients, eComStation and Mainframes, and. since it's a variety of Unix, it merges easily into the high end server environment. On top of this, it's strongly backed by Microsoft's most powerful competitors **(IBM, Oracle, Sun Microsystems)** and even Microsoft's leading allies **(Hewlett Packard, Dell)**.

Microsoft's Response

Microsoft executives are **(bleep)ing bricks** over Linux, and using every method they can to fight it, but with limited success - they were tied up getting convicted on antitrust charges at the critical moment, and it's now hard for them to get the genie back into the bottle. How do you undercut an established product that costs as little as nothing, is highly stable, performs better than your products in most situations, and is more scalable? Bummer!

Evidence of just how desperate Microsoft is, is their blatant financing of SCO's sorry and misguided suit against IBM and Linux, thinly disguised as "licensing Unix". My article SCO - Death Without Dignity describes the whole tawdry affair.

Originally, Microsoft called Linux a **toy**, created by hobbyists for hobbyists. The traumatic end of that line is wonderfully stated in this press conference quote by a top Microsoft executive, "Linux is a toy, well, with IBM backing it, no it's not a toy".

Next was to claim that "Linux is cheap to purchase, but license costs are a very small part of TCO (Total Cost of Ownership)". In truth, other system costs are far greater, making purchase cost almost irrelevant. Microsoft's problem with this line was that many of their customers had already found these other costs to be **lower with Linux** as well.

In mid-July, 2002, at Microsoft's "partner" conference in Los Angeles, Microsoft president Steve Ballmer made an about face. He explained that Linux was **less costly overall,** but that Windows provides "**better value**". This is a more defensible line, since it's impossible to quantify. For some definitions of "value", Windows really can be the better deal, but those definitions are rapidly becoming less numerous.

Then Microsoft launched, through paid writers, Microsoft sponsored lobbying groups, and its own management and sales force, a major **disinformation campaign**, calling Linux and other open source products "**Communist**", "**Un-American**", and "a **major threat to intellectual property and American prosperity**".

Microsoft claimed open source software is not safe to use because it doesn't protect users from possible patent claims. This argument has less impact since Microsoft itself has been shown to have not only **left SQL Server devlopers wide open to patent claims**, but to have deliberately misled them about the risks.

Microsoft claimed open source software is a security risk because the source code is available. Microsoft has argued in court that exposing Windows source code would compromise national security, but they have now **agreed to provide that source code to the government of Russia and China**. Both are not only considered hostile powers by the U.S. government, but are point of origin for many viruses and cyber attacks.

Of all its arguments, Microsoft's **intellectual property claims** sound the most reasonable, but are **completely false**. No one in the world (including entire countries) holds more patents, copyrights and other intellectual property than IBM, yet IBM is a major user and promoter of Linux and other open source products.

Further, any business can incorporate open source code in any way it pleases for its own use **without risk** and without having to reveal anything, so long as it **doesn't distribute the resulting material**. The GPL (General Public License) and other open source licenses limit Microsoft and other software publishers, but not most businesses.

The upshot is, just about nobody (not even technology journalists) was buying Microsoft's disinformation - as confirmed by a captured internal memo.

Then, in December 2002, Microsoft about faced again, saying Windows has **lower TCO** (Total Cost of Ownership) over a 5-year period, and cited an IDC study **which Microsoft paid for**. Problem with the study is, nobody believes it. It not only contradicts established experience, but has flaws obvious even to the untrained eye. Further, one of the authors is on record saying Microsoft's conditions biased the study.

The study presumes: Linux installations need as many administrators as Windows installations (they certainly do not), and Linux administrators cost a lot more than Windows administrators (they do not). Completely omitted are the cost of major Windows upgrades, which are required by License 6, every 3 years. Windows 2000 Server is the 5-year baseline, and it's only been out 2 years, so figures are just projections.

Microsoft is keeping the full study and its methodology secret and releasing only summary figures. This suggests there are many more questionable assumptions and a lot more twisted logic in there.

Other studies have come up with decidedly different results, as did a quantitative analysis of cost, performance and reliability by David Wheeler, and a cost comparison by Robert Frances Group. These results are confirmed by less rigorous studies.

The upshot of all this is that Microsof is resorting to deep discounts and other financial incentives. Recently captured internal memos give the exact amounts available and where they are alocated. See more on this in the next section.

The Foreign Threat

There are many reasons why business and governments outside the United States should be taking a hard look at breaking free of Microsoft, and many are doing so. Security, high cost, ownership of data, balance of trade, and developing a local software industry are prime factors. Several countries have ongoing antitrust action against Microsoft which further encourages them to look at alternatives.

For developing countries, cost is a huge factor. Most have been running on stolen Microsoft software, but international pressure to enforce copyright continues to increase. Many are looking at Linux and other open source products as their best bet - in two ways. Not only does Linux itself promise to save huge amounts of money and help build a local software industry, the Mexican experience has shown the way to use Linux to extort free licenses, and even computer equipment and services, out of Microsoft.

If Microsoft lets Linux take over overseas markets, that's sure to spread. If they bow to extortion, as they have been doing, it also depresses their overseas markets. Though "millions of dollars worth of software" costs Microsoft about $15.00 (the millions only show up on their tax deductions), it means no income is derived from those licenses. In addition, computers to run the software and support services do cost, and these are often part of the deal. Here's a bit of what's going on:

The German Bundeswehr (armed forces) has banned Microsoft products due to real and "suspected" security problems (like little back doors for the NSA, for instance). Note: some have questioned the extent of this ban, since a current Microsoft contract is still in force.
Beyond the Bundeswher, the German government has formulated an ambitious program to move the public sector to open source software. Adoption by various agencies has begun.

Australia government agencies are investigating and adopting Linux at an accelerating rate (O11). Agencies say they stand to save up to 30% on hardware and software. The Department of Veterans Affairs deployed Linux on an IBM zSeries mainframe as a basis for deploying "thin client" workstations, further reducing costs, especially for administration.

Also, Microsoft's largest Australian customer, telephone giant Telstra is considering deploying Linux on 48,000 desktop PCs to reduce costs. They've already chosen Java 2 instead of .NET as their Web Services platform and are considering StarOffice instead of Microsoft Office. Microsoft CEO Steve Ballmer has flown to Australia in hopes of slowing the slide to Linux.

China promised to suppress intellectual property piracy as a condition of entering the WTO (World Trade Organization). China can't afford to buy licenses for the millions of stolen copies of Microsoft software in use, consequently, Linux (particularly the locally produced Red Flag Linux) has been declared the "official" operating system for China. The government is financing Linux development, and is also using the move to Linux to extort cheap licenses from Microsoft.
China has the added incentive of security. It is not always on the best of terms with the U.S. government and is quite confident the CIA has easy entrance to any Windows based software systems.

Now, in August '03, China has announced a formal ban on government agencies buying foreign software, to be effective by the end of 2003. Commercial users are almost sure to follow to maintain compatibility with the government offices they deal with, and it will certainly have an effect on trading partners as well.

Japan is currently completely dominated by Windows. This is becoming a concern for the government, which is now investigating open source software for public use and industry (which includes TRON, a home grown open source operating system). Government officials are strongly emphasizing Linux and TRON development to retain control of its "substantial" consumer electronics industry. Bill Gates has recently flown to Japan in an attempt to disuade government officials from this policy.

Mexico declared it was moving its education system to Linux. Microsoft responded immediately with a lot of free software, some free computers, some free tech support, and a little entertainment for Mexican government officials. The Linux move is on hold, but it won't go away.

The **Peruvian Congress** introduced a bill calling for all government systems to be based on open source software (which includes Linux, but not Windows). Microsoft wrote a letter of protest, to which Peruvian congressman Dr. Edgar Villanueva Nunez wrote his now famous (and totally devastating) response.

Microsoft's reaction was swift. They recruited the U.S. Department of State to apply pressure on Peru, and flew Peruvian President Alejandro Toledo to Redmond Washington where he would be far from his own Congress, They gave him the full tour and brainwashing, an audience with Bill Gates, and sent him home with the gift of $550,000 of Microsoft stuff.

Hmmm . . just $550,000? Other countries have scored a whole lot more. Was some other "consideration" applied to magnify the effectiveness of such a small gift?

Venezuela appears to have rejected the extortion path entirely, and is going straight forward with laws requiring open source software for all government systems. Perhaps Bill Gates' buddy G.W.Bush can arrange a Grenada style invasion or some other "regime change"? Looks like they're working on that already.

England's Prime Minister, Tony Blair, is said to worship Bill Gates as a minor deity. The result has been major contracts handed to Microsoft to develop eGovernment systems. The result has been that only those running the latest Microsoft software have access to government services in England (Duh!).

This has caused considerable backlash, and resulted in new policy cautiously favoring open source software, use of open interoperability standards, and interest in open source initiatives in the European Union.

France has been considering legislation mandating open source software for all government operations. With an economy almost as powerful as California's, France would be difficult for Microsoft to sway with a few software donations, and that would seriously undermine an important revenue stream in any case.

France was the country where Microsoft Office first achieved a true monopoly. Microsoft rewarded the French by doubling the price of the French version of Office and threatening dire consequences to anyone who dared import the much cheaper French Canadian version.

On the other hand, the French worship Jerry Lewis as the king of comedy, so we can't expect their actions to be entirely rational (update: I have received missives from France advising me that many French don't like Jerry Lewis. Encouraging, but please don't send him back).

Sweden's Agency for Public Management has formulated policy based on the principle that "no-one should be forced to use a vendor-specific product in order to communicate with the public sector" (O26). This means across the board adoption of open standards and open source software in government.

Denmark has opted for StarOffice in the schools, and in the homes of students. Both Linux and Windows versions are covered by the agreement with Sun Microsystems.

The **European Union**, aside from separate actions in Germany and France, is strongly considering standardizing all inter-government communications and data systems on open source software. This would seem almost essential to achieve the collaborative goals outlined in the document eEurope 2005: An information society for all.
The seemingly inevitable move of European government to an open source basis would have a strong spill-over effect for all companies doing business with European governments, and for a great many American companies as well.

Republic of South Africa - License 6 was more than the State IT Agency could take, and it's made the decision to convert entirely to open source software. The projected savings is about $333 Million per year. A major consideration was that nearly the entire $333 Million was leaving South Africa every year.

India is faced with issues of security, license costs, and building a local software industry. The government of India decided to heavily promote Linux and open source software throughout the

government and the schools.

Microsoft's response was swift. Bill Gates flew to India and anounced a $100 million contribution to fight AIDS (to be paid out over a number of years). While the AIDS contribution generated positive press, Microsoft contributed $450 million (to be paid immediately) to Indian government agencies to fight Linux. Now that they have the money in hand, India is reportedly going with Linux anyway.

Note: I've recieved an email from a company based in India telling me the $100 million to fight Aids was mostly in Microsoft software to set up Aids information centers, and that the government agencies assistance was also mostly in Microsoft software, so the real cost to Gates / Microsoft was a fraction of the claimed amounts.

California was on the verge of making a deal (Deukmegian administration) to sell its entire education system to Microsoft (as Texas already had done) in exchange for cheap software licenses, but the word got out, there was public outcry, and the deal had to be scrapped, despite campaign contributions from Redmond.
Currently Calfiornia is considering the Digital Software Security Act which favors open source software in government.

Clearly, Microsoft can't continue **bribing all the world** to use Windows, the threat will keep coming back with each upgrade cycle. That $43 billion in the bank just won't stretch that far. Even worse, **American corporations** are starting to learn the **extortion game** too. Rumors abound that if a company demonstrates a strong Linux pilot program, Microsoft sales is authorized to drop license fees by up to 50%.

Now that whole countries can fulfill their basic software needs at little cost through rapidly improving open source products, and can use the open source platform to develop **localized software**, Open source threatens to become the standard for **international commerce**. American import/export business will need to be compatible with their trading partners, and the U.S. military (which already makes use of open source products) will have to be compatible with its allies. The implications for Microsoft's expensive products are clear.

Software Industry Support

In the past, Microsoft found it easy to use extortion and threats against software developers to destroy competing platforms, as they did with OS/2. If software developers won't develop for it, even a vastly superior operating environment is dead.

Today, the situation is a little different. It's the software developers themselves who are clearly targeted for Microsoft destruction. When you have already condemned someone to death, additional threats have diminished impact.

The problem many developers face is their total commitment to Windows and the Windows market. They literally **know nothing else and fear everything else**. They have no choice but to stare into the headlights until impact. Other developers have Unix or Borland Delphi experience, or run their products on standard database engines that have been ported to Linux. These developers have choices for survival, and a few of them will make the right choices to survive.

There are persistent rumors that Microsoft is porting some of its major products to Linux. Microsoft is many things, but **suicidal is not one of them**. Porting would be difficult (the Apple OS X (Unix) version of Office depends on an OS9 translation layer written by Apple), and they certainly wouldn't release it if they did port it. Such an endorsement of the enemy would clearly signal that their **monopolies are smashed**, so expect to see the Devil on ice skates first.

All the major database engines (and many minor ones), with the exception of Microsoft's **SQL Server** and **Access**, have been fully ported to Linux, including **Oracle, IBM's DB2, Informix, Sybase, Pervasive SQL (Btrieve), Advantage, MySQL, Postgres SQL and SAP DB**. Oracle is, in fact, migrating all their in-house applications to Linux.

Sun Microsystems fully supports Linux with all its **Java** products, and has recently authorized development of an open source version of Java. Java is the primary language for developing **Web Services**, and is the environment Microsoft's .NET must compete against.

Accounting Software is very common for Linux. Many accounting vendors had Unix products which were dead easy to port to Linux. Others were still publishing DOS products, also a very easy Linux conversion. Examples are Vigilant, Appgen and Data Pro. Products coded directly for Windows are difficult to port, so those with a "Windows only" product line are unlikely to make the move.

Best's Act! and **Intuit's QuickBooks** are Microsoft's ace in the hole. Until file compatible equivalents are developed, Linux will have a hard time capturing the small business desktop in the U.S.. Both companies are closely tied to Microsoft, so QuickBooks won't be ported until it's too late to save Intuit from Microsoft Great Plains. Symantec's primary business is Norton "fix-it" and anti-virus products specific to Windows. These products **are not needed** for **operating systems that work**, so Symantec will stay committed to the end.

Borland is playing both sides of the fence. They offer strong support for .NET, but also produce leading Java tools and **Kylix,** a rapid development environment for Linux that is largely compatible with their **Delphi** environment for Windows. So important is Borland in providing a path by which Windows developers can move to Linux, rumors abound that Microsoft intends to buy Borland and discontinue most of its products.

Laws and Government

Our "elected representatives" in Washington and State Capitals are elected by the public, but the public is not what they represent - they represent money. Laws recently passed, such as the **DMCA** (Digital Millennium Copyright Act), and laws now under consideration, are not designed to benefit the public, but to benefit specific industries at the expense of the public - industries that contribute heavily to election funds.

Prominent among these contributors are the RIAA (Recording Industry Association of America) and MPAA (Motion Picture Association of America) and Microsoft. These organizations are all able to "influence" as many Congresscritters as they need out of petty cash, and they have placed plenty of items on the legislative plate.

Several proposed laws are aimed at extending corporate ownership and control of intellectual property (books, movies, popular characters, software, etc.) and to reduce the rights and privacy of

individuals. Many have been reworded to include the word "Security", which, since 9/11, has eclipsed "to protect the children" as the all-purpose justification for taking rights away from American citizens.

Several of these acts, both pending and already passed, violate established rights, such as the right of "fair use" (making copies of material you have purchased for your own use, or quoting excerpts for reviews or reference), post-sale rights to use and disposal of property you have paid for, and freedom of speech.

CBDTPA: Not all Congress critters are as outspokenly bought as Fritz Hollings, (D. Disney - oops, I mean South Carolina) and Diane Feinstein (D. California), who are promoting the extremely anti-consumer Consumer Broadband and Digital Television Promotion Act (CBDTPA), but you can expect most to "get with the program".

Even Microsoft has objected to the CBDTPA, not because of the rights it takes from you, but because it would legislate matters of technology that Microsoft feels it alone should control. In other words, the law may not choose Palladium as the means of digital rights control, and has clauses aimed at preventing monopoly.

DMCA: Congress has already passed the DMCA (Digital Millennium Copyright Act, which makes it illegal to make available or to own any device or software that enables you to bypass any means of limiting access (to enable "fair use" for instance). It does include an exemption for cryptographic research, and for "reverse engineering" (to create compatible products), but that definition is very narrow and only exemnpts software, so a method that includes hardware is off limits,

Many consider the DMCA to be very poorly written and wide open to abuse. For example, if a consumer activist Web site offers evidence of corporate wrongdoing by posting incriminating documents, the corporation need only inform the Web site's hosting service that "the site is making unauthorized use of our copyrighted material", and the hosting service is required by law (the DMCA) to immediately take down that material.

There is a protest procedure to get material back up, but it requires the complaining party to immediately file a lawsuit if it wants to keep the material removed, which may discourage many from attempting to get material restored. The site **Chilling Effect Clearinghouse** documents how the DMCA discourages freedom of speech.

Examples: the Church of Scientology has applied the DMCA in an attempt to force Google not to index a site that criticizes Scientology, printer manufacturer Lexmark is using the DMCA to prevent your access to low cost recycled ink and toner cartridges, and Blackboard, a publisher of school administration software, is using it to supress information about the security of its ID card system.

Super DMCA: Not content with the excesses of the DMCA, the MPAA is actively pushing legislation popularly called "Super DMCA" in state legislatures. Super DMCA was written by the MPAA to it's own advantage with total disregard for its broader impact, which is substantial.

This legislation makes it **illegal to protect your network with a firewall**, and makes it **illegal to use a VPN** (Virtual Private Network) to protect your communications with business partners, remote offices and teleworkers. Heavy fines and prison terms are mandated for any attempt to protect your business from hackers and script kiddies.

This legislation also prohibits Internet access for the majority of current business users. There aren't enough "public" IP addresses to go around, and all access from "private" addresses (192.168.n.n for example) is made illegal by Super DMCA because such access can only be from behind a NAT (Network Address Translation) firewall.

Passing this legislation is an act of blind ignorance, yet it has already become law in Michigan, Delaware, Illinois, Maryland and Virginia, and is pending in other states (Maryland and Virginia also passed the infamous UCITA). This is what happens when money comes calling on our legislators, who's average technical expertise apparently extends little beyond operating a light switch.

To avoid legal problems caused by ill-considered laws passed by legislators with no understanding of technology issues, development projects and advocacy Web sites are being moved to other countries, with a net loss of American jobs. It's only going to get worse as "big media" continues to try to solve marketing problems by banning technology.

UCITA: A big Microsoft push is to get UCITA (Uniform Computer Information Transaction Act) passed as an addition to the Uniform Commercial Code all states use as a guide in matters of contract. UCITA makes legally binding "shrink wrap" and "click through" licenses for software, even if you are not allowed to read the license before purchase, and also allows the publisher to change the license terms after acceptance, any time he pleases.

The UCITA "self help" clause gives software publishers a clear legal right to **enter your computer systems** without your knowledge or permission and disable software for any real or imagined violation of license or payment. UCITA also forbids you to disclose license terms to anyone else - if you got screwed, you're not allowed to warn others. Note the similarty between UCITA and clauses in Microsoft's EULA (End User License Agreement) for recent products.

UCITA is opposed even by the **American Bar Association**. The American Library Association, several industry associations, and most State attorneys general also oppose it, but it's supported by Microsoft's money, so it returns from the dead every year, and will do so until it passes.

So far UCITA has passed only in Maryland and Virginia, but that is sufficient to use, as in "This contract falls under the jurisdiction of the laws of the state of Maryland". Do not sign any software contracts that specify laws of Maryland or Virginia.

I suggest reading the references, letting others know what's going on, and writing your "representatives" to let them know that you know what they are up to, and that you don't like it.

Additional Reading

This article has hundreds of references. Please check them out online.

This article is re-printed with permission. The originals can be found at:

www.aaxnet.com/editor/edit029.html

StoreBackup

Author:Heinz-Josef Claes <hjclaes@web.de>

Abstract:

StoreBackup offers itself to the general user who does not neccessarily own a tape backup but a second harddrive or another computer. It offers itself to the users in the professional environment for extremely fast and comfortable access to their backups, also to save on the costs of tapes as well as administrative expenses.

Storage on harddrives or similar devices offers itself as an alternative or additional resource to data backup on tapes. The program to be introduced here performs well and saves storage capacity:

Directories, including their tree structure, may be copied to another location (e.g. /home => /var/bkup/2003.12.13_02.04.26). Permissions to the files remain, enabling users to access the backup directly.

The content of the files is going to be compaired with the existing backup to make sure there is only one backup for each file, that means files with the same content exist physically only once in the backup.

Identical files are hard linked and appear in the backup in the same locations as in the original.
Backup files will be compressed, except they are marked 'exclude'. Compression may be excluded entirely.

Backup series, generated independently (e.g. from different machines) may refer through hard links to shared files. Full or partial backups may be executed with this method, always under the condition that files with the same content may exist only once in the backup.

Why a new backup tool ?

There are possibly thousands of backup programs. So, why another one? The reason arose from my activities as a consultant. The entire week I was moving around and I had no way to secure my data during the week at home. All I had was a 250MB ZIP drive on my parallel port. The backup on the ZIP drive did not give me a lot of storage space and I had to live with a low performance (about 200KB/s). In addition to that I needed fast, simple access to my data - I did not like the usual options of full, differential and incremental backups (e.g. with tar or dump): on one hand it is ususally too cumbersome to retrieve one of the versions, on the other hand it is not possible to delete an old backup at will, this has to be planned carefully at the generation of the backup.

It was my goal to be able to backup quickly during my work and find my files quickly and without hassle.

So, at the end of 1999 the first version of storeBackup was created, it was, however, not suitable for large environments. It was not performing well enough, did not resize sufficiently and was not able do deal with nasty file names (e.g. '\n' in a name).

Based on that experience with the first version I wrote a new one which was published a little bit less than a year later under the GPL. In the meantime the number of users had grown - from home user applications, securing of (mail) directories at ISPs or hospitals as well as universities and for general archiving.

What would be an ideal Backup Tool?

The ideal backup tool would create every day a complete copy of the entire data system (including the applicable access rights) on another data system with minimal effort for the administrator and maximal comfort for the user. The computer and hard disk systems to make this possible should be in a distant, secure building, of course. With the help of a data system browser the user could access the secure data for searching and to copy data directly back. The backup would be usable directly and without problems . Dealing with backups would become something 'normal' - since the route over the administration would in general be unnecessary.

The process described here has a small disadvantage: it needs a lot of harddrive space and it is quite slow because each time the total amount of data needs to be copied.

How does storeBackup work?

StoreBackup tries to accomplish the "ideal backup" and to solve the two problems: storage space and performance.

Features

The first measure to decrease the necessary harddrive storage space would be the compression of data - if that makes sense. storeBackup allows the use of any compression algorithm as an external program. The default is bzip2.

Looking at the stored data closely, it is apparend that from backup to backup relatively few files change - which is the reason for incremental backups. We also find that many files with the same content may be found in a backup because users copy files or a version administration program (like cvs) is activ. In addition, files or directory structures are re-named by users, in incremental backups they are again (unnecessarily) secured. The solution to this is to check the backup for files with the same content (possibly compressed) and to refer to those. The hard link is this reference. (Explanation: data blocks in Unix systems are administered through inodes. Many different file names in as many directories may refer to an inode. The actual file is being deleted with its last hard link (=directory name). (Hard links may point to a specific file only within one file system.)

With this trick of the hard links, which were already created in existing backup files, each file is present in each backup although it exists physically on the harddrive only once. Copying and renaming of files or directories takes only the storage space of the hard links - nearly nothing.

Most likely not only one computer needs to be secured but a number of them. They often have a high proportion of identical files, especially with directories like /etc or /usr. Obviously, there should be only one copy of identical files stored on the backup drive. To mount all directories from the backup server and to backup all computers in one sweep would be the most simple solution. This way duplicate files get detected and hard linked. However, this procedure has the disadvantage that all machines to be secured have to be available for the backup time. That procedure can in many cases not be feasible, for example, if notebooks shall be backed up using storeBackup.

Specifically with notebooks we can find a high overlap rate of files

since users create local copies. In such cases or if servers are backed up independently from one another, and the available harddrive space shall be utilized optimally through hard links, storeBackup is able to hard link files in independent backups (meaning: independent from each other, possibly from different machines).

For the deletion of files storeBackup offers a set of options. It is a great advantage for deletion when each backup is a full backup, those may be deleted indiscriminately. Unlike with traditional backups, there is no need to consider if an incremential backup is depending on previous backups.

The options permit the deletion or saving of backups on specific workdays, first or last day of the week/month or year. It can be assured that a set of a minimum number of backups remains. This is especially useful if backups are not generated on a regular basis. It is possible to keep the last backupsof a laptop until the end of a four week vacation even though the period to keep it is set to three weeks. Furthermore it is possible to define the maximal number of backups. There are more options to resolve the existence of conflicts between contradictory rules (by using common sense).

Performance

The procedure described above assumes that an existing backup is being checked for identical files prior to a new backup of a file. This applies to files in the previous backup as well as to the newly created one. Of course it does not make much sense to directly compare every file to be backed up with the previous backup. So, the md5 sums of the previous backup are being compared with the md5 sum of the file to be backed up with the utilization of the hash table. The program is using dbm files for this.

Computing the md5 sum is fast, but in case of a large amount of data is still not fast enough. For this reason storeBackup checks initially if the file was altered since the last backup (path + file name, ctime, mtime and size are the same). If that is the case, the md5 sum of the last backup is being adopted and the hard link set. If the initial check shows a difference, the md5 sum is being computed and a check takes place to see if another file with the same md5 sum exists. (The comparison with a number of backup series uses a expanded but similarily efficient process). For this approach only a few md5 sums need to be calculated for a backup.

My server (200 MHz, IDE) processes about 20 to 35 files/second, my desktop machine (800MHz,IDE) about 150 to 200 files/second. On fast computers with fast harddrives (2.4 GHz, 1.4TB software RAID) I have measured 800 files/second. These results are for writing to local drives. Writing over NFS gets is a lot slower. Crucial is the speed of the harddrive. (All tests were done under Linux).

Implementations

The storeBackup tools have been testet on Linux, FreeBSD, Solaris and AIX. They should be able to run on all Unix plattforms. Perl was used as the programming language.

Installation

The installation is simple. StoreBackup can be downloaded from http://www.sf.net/projects/storebackup as storeBackup version.tar.bz2 and unpacked to the desired location.

```
tar jxf storeBackup-version.tar.bz2
```

This creates the directory storeBackup with the documentation and the executables in the subdirectory bin. They can be called with the complete path. As an alternative the $PATH environment variable may be set. Operating systems which do not have the program md5sum included (e.g. FreeBSD) need to compile it. Instructions for this can be found in the attached README file.

Operation

We shall not describe all options here in detail, that can be found in the software package.

The simplest method to generate a backup is:

```
storeBackup.pl -s sourceDir -t targetDir
```

sourceDir und targetDir must be existing. StoreBackup will copy the files from sourceDir to targetDir/date_time and in this procedure compressing them with bzip2 (avoiding .gz, bz2, .png etc) as well as linking duplicate files.

In its up- to- date version (1.14.1) storeBackup.pl has 45 parameter at its disposal, to describe them here would go beyond the scope of this article. They can be accessed with

```
storeBackup.pl -h
```

In the files README and EXAMPLES we can find exhaustive explanations on the different applications. It shall be pointed out that the alternative to putting the parameters in the command line - which can become complex quickly - a configurations file may be used. It can be generated with

```
storeBackup.pl --generate --file ConfigFile
```

or shorter with

```
storeBackup.pl -g -f ConfigFile
```

After finalising the configuration it may be read, the syntax checked and partially applied by

```
storeBackup.pl -f ConfigFile --print
```

subsequently storeBackup may be startet with

```
storeBackup.pl -f ConfigFile
```

The entire description of all options of storeBackup can be found in the files README and EXAMPLES which are part of the tar file.

To detect where which version of a file in a backup exists, storeBackup can be utilized:

```
storeBackupVersion.pl -f Filename
```

filename is the name of the file in question, it has to be written just like it is in the backup, i.e. with its compression attributes. To go to the backup directory in the correct location and executing the command is the easiest way. Exercising the option "-h" will exhibit explanations to all 11 parameter.

The recovery of single files may be done with cp, ftp, file browser or similar mechanism. For the recovery of partial directory trees or complete backups it makes sense to use the applicable tool storeBackupRecover.pl It will extract the wanted files or directories from the backup. This will restore the original, i.e. user, group and rights will be re-established. The files will also be decompressed if they were so in the original version. Original hard links will be restored too.

Additional options in storeBackup permit statistical readouts, like the manipulation of performance parameters, the overwrite behaviour and others. A total of 10 parameters may be read out by using the option "-h".

With storeBackupDel.pl backups may be deleted independently from the program storeBackupRecover.pl. This can be useful in case of a backup over NFS. Deleting directory trees over NFS is much slower than local deletion. storeBackup may be called over the NFS without delete function, this allows a better control the backup duration. The deletion of previously generated backups on the server with storeBackupDel - which, by the way, has the same options for the deletion as storeBackup - can be decoupled from the actual backup process.

Existing backups are organized in directories. They can be displayed with storeBackupls.pl (more coherent than with 'ls'). Simpy as a list

```
[ACL tag]:[ACL qualifier]:[Access permissions]
hjc@schlappix:~/backup ) storeBackupls.pl /
media/zip/stbu/
  1  Fri May 23 2003   2003.05.23_12.37.53   -156
  2  Fri Jun 06 2003   2003.06.06_14.31.47   -142
  3  Fri Jun 13 2003   2003.06.13_14.17.18   -135
  4  Fri Jun 20 2003   2003.06.20_14.02.35   -128
  5  Fri Jun 27 2003   2003.06.27_14.23.55   -121
  6  Mon Jun 30 2003   2003.06.30_17.34.37   -118
  7  Fri Jul 04 2003   2003.07.04_13.10.06   -114
  8  Fri Jul 11 2003   2003.07.11_13.13.14   -107
  9  Fri Jul 18 2003   2003.07.18_14.03.49   -100
 10  Fri Jul 25 2003   2003.07.25_14.19.19   -93
 11  Thu Jul 31 2003   2003.07.31_17.07.55   -87
 12  Fri Aug 01 2003   2003.08.01_12.16.58   -86
 13  Fri Aug 15 2003   2003.08.15_15.10.19   -72
 14  Sat Aug 23 2003   2003.08.23_06.25.35   -64
 15  Wed Aug 27 2003   2003.08.27_18.21.09   -60
 16  Thu Aug 28 2003   2003.08.28_14.16.39   -59
 17  Fri Aug 29 2003   2003.08.29_14.35.10   -58
 18  Mon Sep 01 2003   2003.09.01_17.19.56   -55
 19  Tue Sep 02 2003   2003.09.02_18.18.46   -54
 20  Wed Sep 03 2003   2003.09.03_16.22.41   -53
 21  Thu Sep 04 2003   2003.09.04_16.59.19   -52
 22  Fri Sep 05 2003   2003.09.05_14.35.20   -51
 23  Mon Sep 08 2003   2003.09.08_20.08.52   -48
 24  Tue Sep 09 2003   2003.09.09_18.45.48   -47
 25  Wed Sep 10 2003   2003.09.10_18.30.48   -46
 26  Thu Sep 11 2003   2003.09.11_17.26.46   -45
 27  Fri Sep 12 2003   2003.09.12_15.23.03   -44
 28  Mon Sep 15 2003   2003.09.15_18.05.19   -41
 29  Tue Sep 16 2003   2003.09.16_18.04.16   -40
 30  Wed Sep 17 2003   2003.09.17_19.03.02   -39
 31  Thu Sep 18 2003   2003.09.18_18.21.09   -38
 32  Fri Sep 19 2003   2003.09.19_14.48.05   -37
not finished
 33  Mon Sep 22 2003   2003.09.22_18.58.55   -34
 34  Tue Sep 23 2003   2003.09.23_18.48.40   -33
 35  Wed Sep 24 2003   2003.09.24_19.32.24   -32
 36  Thu Sep 25 2003   2003.09.25_18.05.38   -31
 37  Fri Sep 26 2003   2003.09.26_14.59.59   -30
 38  Mon Sep 29 2003   2003.09.29_18.42.59   -27
 39  Tue Sep 30 2003   2003.09.30_18.02.03   -26
 40  Wed Oct 01 2003   2003.10.01_17.09.43   -25
 41  Thu Oct 02 2003   2003.10.02_15.26.33   -24
 42  Mon Oct 06 2003   2003.10.06_20.08.45   -20
 43  Tue Oct 07 2003   2003.10.07_19.46.54   -19
 44  Wed Oct 08 2003   2003.10.08_16.03.23   -18
 45  Thu Oct 09 2003   2003.10.09_16.58.28   -17
 46  Fri Oct 10 2003   2003.10.10_14.21.06   -16
 47  Mon Oct 13 2003   2003.10.13_18.58.24   -13
 48  Tue Oct 14 2003   2003.10.14_16.02.44   -12
 49  Wed Oct 15 2003   2003.10.15_19.04.12   -11
 50  Thu Oct 16 2003   2003.10.16_15.47.51   -10
 51  Mon Oct 20 2003   2003.10.20_09.34.52   -6
 52  Mon Oct 20 2003   2003.10.20_12.16.40   -6
 53  Tue Oct 21 2003   2003.10.21_09.43.40   -5
 54  Tue Oct 21 2003   2003.10.21_11.22.36   -5
 55  Tue Oct 21 2003   2003.10.21_16.01.15   -5
 56  Tue Oct 21 2003   2003.10.21_18.08.07   -5
 57  Wed Oct 22 2003   2003.10.22_10.02.51   -4
 58  Wed Oct 22 2003   2003.10.22_16.09.42   -4
 59  Wed Oct 22 2003   2003.10.22_18.03.05   -4
 60  Thu Oct 23 2003   2003.10.23_08.18.15   -3
 61  Thu Oct 23 2003   2003.10.23_14.16.24   -3
 62  Thu Oct 23 2003   2003.10.23_17.00.36   -3
 63  Fri Oct 24 2003   2003.10.24_13.29.30   -2
 64  Sun Oct 26 2003   2003.10.26_09.08.55   0
```

'not finished' means the backup was abortet). or with information on the deletion conditions in the configuration file:

```
hjc@schlappix:~/backup ) storeBackupls.pl -f
stbu.conf /media/zip/stbu/
analyse of old Backups in </media/zip/stbu/>:
 Fri 2003.05.23_12.37.53 (156): keepLastOfMonth
(400d)
 Fri 2003.06.06_14.31.47 (142): keepLastOfWeek
(150d)
 Fri 2003.06.13_14.17.18 (135): keepLastOfWeek
(150d)
 Fri 2003.06.20_14.02.35 (128): keepLastOfWeek
(150d)
 Fri 2003.06.27_14.23.55 (121): keepLastOfWeek
(150d)
 Mon 2003.06.30_17.34.37 (118): keepLastOfMonth
(400d)
 Fri 2003.07.04_13.10.06 (114): keepLastOfWeek
(150d), keepMinNumber50
 Fri 2003.07.11_13.13.14 (107): keepLastOfWeek
(150d), keepMinNumber49
 Fri 2003.07.18_14.03.49 (100): keepLastOfWeek
(150d), keepMinNumber48
 Fri 2003.07.25_14.19.19 (93): keepLastOfWeek
(150d), keepMinNumber47
 Thu 2003.07.31_17.07.55 (87): keepLastOfMonth
(400d), keepMinNumber46
 Fri 2003.08.01_12.16.58 (86): keepLastOfWeek
(150d), keepMinNumber45
 Fri 2003.08.15_15.10.19 (72): keepLastOfWeek
(150d), keepMinNumber44
 Sat 2003.08.23_06.25.35 (64): keepLastOfWeek
(150d), keepMinNumber43
 Wed 2003.08.27_18.21.09 (60): keepMinNumber42,
keepWeekDays(60d)
 Thu 2003.08.28_14.16.39 (59): keepMinNumber41,
keepWeekDays(60d)
 Fri 2003.08.29_14.35.10 (58): keepLastOfMonth
(400d), keepLastOfWeek(150d),
                                keepMinNumber40,
keepWeekDays(60d)
 Mon 2003.09.01_17.19.56 (55): keepMinNumber39,
keepWeekDays(60d)
 Tue 2003.09.02_18.18.46 (54): keepMinNumber38,
keepWeekDays(60d)
 Wed 2003.09.03_16.22.41 (53): keepMinNumber37,
keepWeekDays(60d)
 Thu 2003.09.04_16.59.19 (52): keepMinNumber36,
keepWeekDays(60d)
 Fri 2003.09.05_14.35.20 (51): keepLastOfWeek
(150d), keepMinNumber35, keepWeekDays(60d)
 Mon 2003.09.08_20.08.52 (48): keepMinNumber34,
keepWeekDays(60d)
 Tue 2003.09.09_18.45.48 (47): keepMinNumber33,
keepWeekDays(60d)
 Wed 2003.09.10_18.30.48 (46): keepMinNumber32,
keepWeekDays(60d)
 Thu 2003.09.11_17.26.46 (45): keepMinNumber31,
keepWeekDays(60d)
 Fri 2003.09.12_15.23.03 (44): keepLastOfWeek
(150d), keepMinNumber30, keepWeekDays(60d)
 Mon 2003.09.15_18.05.19 (41): keepMinNumber29,
keepWeekDays(60d)
 Tue 2003.09.16_18.04.16 (40): keepMinNumber28,
keepWeekDays(60d)
 Wed 2003.09.17_19.03.02 (39): keepMinNumber27,
keepWeekDays(60d)
 Thu 2003.09.18_18.21.09 (38): keepMinNumber26,
keepWeekDays(60d)
 Fri 2003.09.19_14.48.05 (37): keepLastOfWeek
(150d), keepMinNumber25, keepWeekDays(60d)
 Mon 2003.09.22_18.58.55 (34): keepMinNumber24,
keepWeekDays(60d)
 Tue 2003.09.23_18.48.40 (33): keepMinNumber23,
keepWeekDays(60d)
 Wed 2003.09.24_19.32.24 (32): keepMinNumber22,
keepWeekDays(60d)
 Thu 2003.09.25_18.05.38 (31): keepMinNumber21,
keepWeekDays(60d)
 Fri 2003.09.26_14.59.59 (30): keepLastOfWeek
(150d), keepMinNumber20, keepWeekDays(60d)
 Mon 2003.09.29_18.42.59 (27): keepMinNumber19,
```

```
keepWeekDays(60d)
 Tue 2003.09.30_18.02.03 (26): keepLastOfMonth
(400d), keepMinNumber18, keepWeekDays(60d)
 Wed 2003.10.01_17.09.43 (25): keepMinNumber17,
keepWeekDays(60d)
 Thu 2003.10.02_15.26.33 (24): keepLastOfWeek
(150d), keepMinNumber16, keepWeekDays(60d)
 Mon 2003.10.06_20.08.45 (20): keepMinNumber15,
keepWeekDays(60d)
 Tue 2003.10.07_19.46.54 (19): keepMinNumber14,
keepWeekDays(60d)
 Wed 2003.10.08_16.03.23 (18): keepMinNumber13,
keepWeekDays(60d)
 Thu 2003.10.09_16.58.28 (17): keepMinNumber12,
keepWeekDays(60d)
 Fri 2003.10.10_14.21.06 (16): keepLastOfWeek
(150d), keepMinNumber11, keepWeekDays(60d)
 Mon 2003.10.13_18.58.24 (13): keepMinNumber10,
keepWeekDays(60d)
 Tue 2003.10.14_16.02.44 (12): keepMinNumber9,
keepWeekDays(60d)
 Wed 2003.10.15_19.04.12 (11): keepMinNumber8,
keepWeekDays(60d)
 Thu 2003.10.16_15.47.51 (10): keepLastOfWeek
(150d), keepMinNumber7, keepWeekDays(60d)
 Mon 2003.10.20_09.34.52 (6): keepDuplicate(7d)
 Mon 2003.10.20_12.16.40 (6): keepMinNumber6,
keepWeekDays(60d)
 Tue 2003.10.21_09.43.40 (5): keepDuplicate(7d)
 Tue 2003.10.21_11.22.36 (5): keepDuplicate(7d)
 Tue 2003.10.21_16.01.15 (5): keepDuplicate(7d)
 Tue 2003.10.21_18.08.07 (5): keepMinNumber5,
keepWeekDays(60d)
 Wed 2003.10.22_10.02.51 (4): keepDuplicate(7d)
 Wed 2003.10.22_16.09.42 (4): keepDuplicate(7d)
 Wed 2003.10.22_18.03.05 (4): keepMinNumber4,
keepWeekDays(60d)
 Thu 2003.10.23_08.18.15 (3): keepDuplicate(7d)
 Thu 2003.10.23_14.16.24 (3): keepDuplicate(7d)
 Thu 2003.10.23_17.00.36 (3): keepMinNumber3,
keepWeekDays(60d)
 Fri 2003.10.24_13.29.30 (2): keepLastOfWeek(150d),
keepMinNumber2, keepWeekDays(60d)
 Sun 2003.10.26_09.08.55 (0): keepLastOfMonth
(400d), keepLastOfWeek(150d),
                              keepMinNumber1,
keepWeekDays(60d)
```

In addition to the backup program described above the programs llt and multtail are present. llt will generate the display of the times for creating-, modifying- and access time of files. multitail allows tracking of a number of files like using 'tail-f" but multitail offers more options than 'tail-f' and it is more robust.

Future Plans

For the next versions of storeBackup the following features are planned:

The worst time consumer of a backup (except the first backup during which everything gets compressed/ copied) is the hard linking. To generate a hard link is fast, but due to their large number - compared to the other operations and the parallel operations for compression specifically - this is the main time demand.

The next version of storeBackup will offer the option to backup the directory structure and modified files in a first step. This concludes the backup from the view of the data to be secured. In a second step the missing hard links will be created. These two steps will be completely disconnected from each other - meaning they can be run on different machines and it will be feasible to do several backups prior to generating new hard links.

Initial measurements indicate this option will result in a performance gain - compared to the "normal" full backup - by a factor of 5-10 (1/5 to 1/10 of the "normal"), if local writing is executed. Backup up over the NFS will be much faster if you start the process for hard linking locally on the remote machine.

The plan for the next version will be the expansion of the search capabilities (with subsequent re-backup). It shall be possible to search the backups with a user-defined rule consisting of file name (pattern), file size, time of initial generation/ change, user i.d., group i.d., access rights on the file and a (simple) grep. The rules will include 'and', 'or', 'not' and optional parantheses.
Subsequent future plans envision an expansion of the options (in a tar-like fashion) and the support of new data types, e.g. devices.

Version and License

At the writing of this article the current version of storeBackup is 1.14.1. to be found at http://www.sf.net/projects/storebackup for downloading.
StoreBackup is covered by the GPL.

This article is re-printed with permission. The originals can be found at:

http://www.linuxfocus.org/English/January2004/article321.shtml

AUUG Corporate Members

as at 1st March 2004

- ac3
- Apple Computer Australia Pty Ltd
- Australian Taxation Office
- BAE Systems
- Cape Grim B.A.P.S
- Computer Associates
- Corinthian Industries (Holdings) Pty Ltd
- CSIRO Manufacturing Science and Technology
- Curtin University of Technology
- Deakin University
- Department of Land & Water Conservation
- Department of Lands
- Everything Linux & Linux Help
- EWA-Australia Pty Ltd
- IBM
- IBM Linux Technology Centre
- IP Australia
- KAZ Technology Services
- LPINSW
- Macquarie University
- Multibase WebAustralis Pty Limited
- NSW Dept of Commerce
- Peter Harding & Associates Pty. Ltd.
- Powerhouse Museum
- Squiz Pty Ltd
- Sun Microsystems
- Sydney Water Corporation
- Tellurian Pty. Ltd.
- The University of Western Australia
- Thiess Pty Ltd
- TMD Computing
- Uni of NSW - Computer Science & Engineering
- UNiTAB Limited
- University of New England
- University of New South Wales
- University of Sydney
- University of Technology, Sydney
- University of Technology, Sydney
- Workcover Queensland

Programmer's Toolkit: Profiling programs using gprof

Author: Vinanyak Hegde <vinayak@myrealbox.com>

Introduction to the Series

Linux (and other Unices) have lots of nifty small utilities which can be combined together to do interesting things. There is a certain joy in creating these software or using them to tweak your programs. In this series we shall look at some such tools which are useful for a programmer. This tools will help you to code better and make your life easy.

What is Profiling ?? Why you need it ??

After we have designed and coded a software comes the stage of optimizing the program. Before we talk about profiling and optimization in general I would like to draw your attention to two quotes regarding optimization.

- More computing sins are committed in the name of efficiency (without necessarily achieving it) than for any other single reason - including blind stupidity.
  -- William A. Wulf
- We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil.
  -- Donald E. Knuth

Most programs roughly follow what is known as the 80:20 rule. You will be executing 20% of the code 80% of the time. As is implied by the quotes above programmer time is more valuable than machine time. So we have seen the rise of languages such as Java and C# which reduce time needed to program giving programmers more time to concentrate on the logic rather than the nitty-gritties of the underlying machine architecture. This has increased the running time of the programs but saved programmer time. However we need to optimize to make a program run faster. Many time compilers do this automatically. For example the GCC compiler has the -O (note the upper case) flags to specify the level of optimization. Profiling is a method which can help us to find which sections of code/function we need to optimize to increase the performance of a program. You will agree that it makes a lot more sense to optimize a function which is called thousand times when a program runs rather than one which is called ten times in a program. When we profile a program we will come to know which parts of the code are frequently used and which functions take up the most CPU time. Both of these are good candidates for optimization. Since this data is collected using an actual execution trace, it is also a good method for finding hidden bugs. You may not expect a certain function to be called 1000 times during the execution so this might be defect in the design and a potential bug. This is almost as useful as code reviews in large and complex projects.

There are mainly 2 types of profiling information we can get :-

- **Flat Profile**
  The flat profile details how much CPU time each function used up and the number of times it was called. This is the brief summary of the profiling information gathered. This will give an idea of which functions can be rewritten or tweaked to get performance benefits.
- **Call Graph**
  The call graph shows for every function in the code the number of times it was called by different functions including itself. This can suggest which function calls can be eliminated or replaced by other efficient functions. This information reveals the interrelations between different functions and can be used to uncover bugs in the code. Also you may want to optimize certain code paths after looking at the call graphs.

How to gather profiling information ??

The source code has to be compiled with the -pg option (also with -g if you want line-by-line profiling). If the number of lines in the Make file is small you can append these options to each compilation command. However if the number of compilation commands is large then you can define/redefine the CFLAGS/CXXFLAGS parameter in the makefile and add this to every compilation command in the makefile. I will demonstrate the use of gprof using the gnu make utility.

Unpack the gzipped tarball

```
$ tar zxf make-3.80.tar.gz
$ cd make-3.80
```

Run the configure script to create the makefiles

```
$ ./configure
```

[configure output snipped]

Edit the CFLAGS parameter in the makefile generated to remove optimization flags and add -pg to CFLAGS. GCC optimization flags are removed as compiler optimization can sometimes cause problems while profiling. Especially if you are doing line-by-line profiling, certain lines may be removed while optimizing source code.

Build the source code

```
$ make
```

[build output snipped]

We can use this make to build other software such as Apache, lynx and cvs. We build apache using this make as an example. When we untar, configure and run make on the source of Apache , a file called gmon.out containing profiling information is generated. You may observe that make may run slower than expected as it is logging the profile data. An important thing to be remembered while collecting profile data is that we have to run the program giving it the inputs we give it normally and then exiting when it is all done. This way you would have simulated a real-world scenario to collect data.

Analyzing profiling output

In the last step we have got a binary output file called "gmon.out". Unfortunately there is no way currently to specify the name for the profiling data file. This "gmon.out" file can be interpreted by gprof to generate human readable output. The syntax for the same is :

```
gprof options [Executable file [profile data \
   files ... ] ] [ > human-readable-output-file]
```

```
$ gprof make gmon.out > \
       profile-make-with-Apache.txt
```

you can find the whole file
http://linuxgazette.net/100/misc/vinayak/profile-make-with-Apache.txt

A section of the flat profile is shown below -

```
Flat profile:
```

Each sample counts as 0.01 seconds.

```
  %   cumulative   self              self     total
 time   seconds   seconds    calls  ms/call  ms/call  name
 33.33      0.01     0.01      207     0.05     0.05  file_hash_2
 33.33      0.02     0.01       38     0.26     0.26
new_pattern_rule
 33.33      0.03     0.01        6     1.67     2.81  pattern_search
  0.00      0.03     0.00     2881     0.00     0.00  hash_find_slot
  0.00      0.03     0.00     2529     0.00     0.00  xmalloc
  0.00      0.03     0.00     1327     0.00     0.00  hash_find_item
  0.00      0.03     0.00     1015     0.00     0.00
directory_hash_cmp
  0.00      0.03     0.00      963     0.00     0.00
find_char_unquote
  0.00      0.03     0.00      881     0.00     0.00  file_hash_1
    0.0 0.03     0.00      870     0.00 0.00  variable_buffer_output
```

From the above data we can draw the following conclusions :

1. 3 functions (file_hash_2, new_pattern_rule and pattern_search) take almost all of the time.
2. There are 6 function calls to pattern_search but takes up an average of 2.81 milliseconds for each call.

This is however insufficient data for gathering information. So this specially compiled make was used for building lynx, cvs, make and patch. All the renamed gmon.out files were gathered and profiling data was compiled using the following commands.

```
$ gprof make gmon-*.out > overall-profile.txt
```

This file can be found
http://linuxgazette.net/100/misc/vinayak/overall-profile.txt

A section of the flat profile section is shown below.

```
Flat profile:
```

Each sample counts as 0.01 seconds.

```
  %   cumulative   self              self     total
 time   seconds   seconds    calls  ms/call  ms/call  name
 18.18      0.06     0.06    23480     0.00     0.00
find_char_unquote
 12.12      0.10     0.04      120     0.33     0.73  pattern_search
  9.09      0.13     0.03     5120     0.01     0.01
collapse_continuations
  9.09      0.16     0.03      148     0.20     0.88  update_file_1
  9.09      0.19     0.03       37     0.81     4.76  eval
  6.06      0.21     0.02    12484     0.00     0.00  file_hash_1
  6.06      0.23     0.02     6596     0.00     0.00  get_next_mword
  3.03      0.24     0.01    29981     0.00     0.00  hash_find_slot
  3.03      0.25     0.01    14769     0.00     0.00  next_token
    3.3 0.26     0.01     5800     0.00 0.00  variable_expand_string
```

As we can see, the picture has changed a bit from the make profile we got from compiling apache.

1. There are 23480 calls to the function find_char_unquote and it makes up more than 1/6th of the program execution time.
2. However the function eval has only 37 invocations o it's credit still it takes up about 1/11th of the program execution time. There is a possibility that this function is doing a lot of work and is a candidate for splitting up into different functions. Also notice that each call to eval eats up an average of 4.76 milliseconds which is quite huge compared to any of the other functions
3. Also the functions pattern_search and update_file_1 take up nearly 1/4th of the execution time but share only 268 calls between them. Maybe these functions can also be split into smaller functions.

Let us now have a look at a snippet of the call graph profile from compiling Apache.

```
index % time    self  children    called        name
-----------------------------------------------
                                     6             eval_makefile  [49]
[25]     3.7    0.00    0.00       6         eval  [25]
                0.00    0.00     219/219
try_variable_definition [28]
                0.00    0.00      48/48          record_files [40]
                0.00    0.00     122/314
variable_expand_string  [59]
                0.00    0.00       5/314
allocated_variable_expand_for_file  [85]
                0.00    0.00     490/490         readline [76]
                0.00    0.00     403/403
collapse_continuations [79]
                0.00    0.00     355/355         remove_comments
[80]
                0.00    0.00     321/963         find_char_unquote
[66]
                0.00    0.00     170/170         get_next_mword [88]
                0.00    0.00     101/111         parse_file_seq [93]
                0.00    0.00     101/111         multi_glob [92]
                0.00    0.00      48/767         next_token [70]
                0.00    0.00      19/870
variable_buffer_output [68]
                0.00    0.00      13/2529        xmalloc [64]
                0.00    0.00       2/25          xrealloc [99]
                                   5             eval_makefile  [49]
```

We can make the following observations from the snippet above :

1. The first column gives an index into the function index which is printed at the end of gprof's output.
2. The second column gives the total amount of time spent in the function eval including it's calls to other functions.
3. The third and the fourth column give the total amount of time which is spent in the function itself and call to other functions
4. The first number in the fifth column gives the number of calls to the function from eval and the second number in the column gives the total number of non-recursive calls to that function from all callers.
5. If there are recursive calls from the function to itself or to a mutually recursive function, then the name of the function is appended with cycle (as in eval_makefile and eval above).
6. Some of the functions are called always from eval. It might be advantages in some cases if the overhead of the function call itself is eliminated.

Other gprof facilities

Using gprof you can also get annotated source list and line-by-line profiling. These might be useful once you have identified the the sections of code that need to be optimized. These options will help you drill down in the source code to find inefficiencies. Line-by-line profiling along with flat profile can be used to check which are the code paths which are frequently traversed. The annotated source listing can be used to drill down within function calls themselves up to the basic block (loops and branching statements), to find out which loops are executed most and which branches are taken most frequently . This is useful in fine tuning the code for optimum performance. There are some other options which are not covered here. Refer to the info documentation of gprof for more details. There is a KDE front end which is available for gprof called kprof. See the reference section for the URL.

Conclusion

Profiling tools such as gprof can be a big help in optimizing programs. Profiling is one of the first steps for manual optimization of programs to know where the bottlenecks and remove them.

Resources

- The GNU Info document for gprof
- The KDE front end for gprof http://kprof.sourceforge.net
- Function Check - another profiling tool This overcomes some deficiencies of gprof

Vinayak is currently pursuing the APGDST course at NCST. His areas of interest are networking, parallel computing systems and programming languages. He believes that Linux will do to the software industry what the invention of printing press did to the world of science and literature. In his non-existent free time he likes listening to music and reading books. He is currently working on Project LIberatioN-UX where he makes affordable computing on Linux accessible for academia/corporates by configuring remote boot stations (Thin Clients).

This article is re-printed with permission. The originals can be found at:

http://linuxgazette.net/100/vinayak.html

Certs for the Masses

The Case for a Community-Oriented Certificate Authority

Author: ©2004 Adam Butler <adam@donkeyrequiem.com>

Secure authentication and encryption methodologies want to be free.

Okay, I admit it. Compared with all the other OSS anthropomorphisms floating around, that one's a bit of a mouthful. Nevertheless, the need for strong and reliable data security is as old as data itself.

While the Internet community has championed the "information wants to be free" cause for as long as I can remember, this concept has always been tempered with a profound respect for personal privacy. Consistently, the heroes of the open source movement trumpet the emancipation of innumerable ones and zeroes across the globe while contemporaneously applauding the individual's right to keep his or her ones and zeroes private and secure.

Savvy computer users recognised this need from the very beginning not because they had anything in particular to hide; rather, they merely realised that private data wasn't safe from prying eyes unless specific steps were taken to ensure that safety.

Long before buggy WEP-encrypted WLAN access points dotted the landscape—hell, even before the 1990s Internet retailing explosion —countless individuals sent countless petabytes of God-knows-what to God-knows-who without realising that every bit of their communications could be (and often were) intercepted by others.

Over time, folks wised up. For the sysadmins among us, ask yourself: When was the last time you accessed one of your boxes in an open, untrusted environment, using telnet rather than SSH?

And even Joe User caught on, eventually learning to check his browser for that nifty lock/key icon before submitting his online purchase. Sure, he probably still has little or no idea what is meant by terms like "Secure Sockets Layer" or "128-bit encryption," but at least he knows to check first before spiriting his credit card information off into the ether as clear text.

I doubt anyone would seriously discount the role of PKI, SSL, et al, in strengthening consumer confidence in secure web transactions and thereby laying the groundwork that allowed companies like Amazon and eBay to succeed—but the Public Key Infrastructure allows for so much more than mere virtual mercantilism.

For the most part, the Internet community exploits only a tiny fraction of what this valuable technology has to offer—and with gross privacy violations occurring at disturbingly increasing frequencies (1), it would seem that now more than ever, the importance of publicly available cryptography tools and techniques cannot be understated.

It's time to take the next steps in securing our personal data and that of our users. For that, we're going to need a Certificate Authority.

Enter CAcert

Until recently, the thought of approaching a CA for not one but *numerous* X.509 certificates might have tied your stomach in knots, caused you to break out in hives, and may have even prompted you to murder your entire family. Because unless Daddy's trust fund left you so much dough that you're routinely torching $100 bills just to light your Havanas, you're probably turned off a bit by the realisation that the best price any CA offers is still going to require you take out a second mortgage on the house.

But Dylan quotes so often lend themselves to the OSS movement, and now is no exception: Times are indeed a-changin'.

Late last year, CAcert, a nonprofit, OSS-based Certificate Authority quietly stepped forward with a proposal that was as simple as it was groundbreaking: the Australian-borne organisation would offer signed, 128-bit X.509 certificates for personal and server-side use… for free.

Like so many open source mavericks before them, a small group of committed individuals simply took a long, hard look at a particular industry—in this case, the buying and selling of X.509 certificates —and realised they could do a better job. In almost no time at all, CAcert was providing gratis what industry leaders Thawte and VeriSign were routinely hawking for hundreds or even thousands of dollars apiece.(2)

Today, CAcert offers signed, 128-bit X.509(v3) certificates for SSL, Wireless Auth, S/MIME, VPN, and other authentication/encryption schemes. And whether you're in the market for a personal or server-side solution, you can leave your cache of Spanish doubloons at home—CAcert's expenses are still covered by donations and advertising, not exorbitant (and unnecessary) annual fees.

And that's not all. The venerable CA already offers a highly thought-out "Web of Trust" assurance scheme (3), gently lifted from the highly thought-out WOT scheme offered by Thawte, (4) which was in turn borrowed from the highly thought-out WOT scheme developed by Phil Zimmerman and the folks at PGP(5) The WOT program allows CAcert's more than 5000 members to notarise/sign/assure (depending on whose terminology you prefer) one another in pursuit of "Trust Points."

As a user increases his or her number of trust points with CAcert, advanced features are unlocked and become available for use. One such feature allows users to submit their PGP/GPG key to be signed by the CAcert master key, a novel integration of multiple PKI technologies.

Another feature, expected to be in place by the time you read this, will be the availability of so-called "code signing" certificates—similar in concept to those used in Microsoft's Authenticode initiative,(1) but minus the evil. CAcert sees this as a chance to give back to its fellow open source compatriots, empowering developers on various OSS projects with the means to digitally sign their work without having to rely on certs from expensive, corporate CAs who

could care less about the OSS community.

Supporting the OSS Infrastructure

Undoubtedly the most important role of a Community-Oriented Certificate Authority is to provide an affordable alternative to commercial certificate authorities, thus enabling thousands of smaller web presences to abandon their current hackneyed PKI implementations and fall under the umbrella of a true CA, rather than relying on self-generated certificates in which users are (rightfully) leery of placing their trust.

As the situation currently stands, webmasters who wish to employ some type of Public Key Infrastructure—SSL, for example—usually feel that they must choose between (1) paying hundreds of dollars each year for a "trusted" certificate signed by some big name CA, or (2) grabbing a current copy of the SSL libraries and generating their own self-signed, "untrusted" cert for $0. Unsurprisingly, many of these webmasters opt for the second choice—necessitating that each of their (apparently *quite* trusting) users download and install their sites' home-brewed root certificates, always assuming/trusting that Webmaster X **really is** Webmaster X, even if no one has ever confirmed this in any form or fashion.

With CAcert, a new option unfolds. Rather than fool around with generating a homebrew SSL cert, a webmaster unwilling to pony up for commercial certificate services can instead obtain one signed by CAcert. And unlike the self-signed certificate, CAcert "vouches for" its certificate and reveals to site visitors (via trust points) how well known/trusted the webmaster is by the CA, giving visitors to the site straightforward, independent verification that Bob's Porn Palace is indeed operated by Bob.

Additionally, as more webmasters abandon self-signed certificates for flexible, widely-available CAcert products, they free themselves of having to publish site-specific root certificates, revocation lists, and the like. Users simply install CAcert's root certificate—which isn't that much to ask, considering that CAcert (as an independent CA) employs the same methods of member verification as its for-profit competitors—and voila, they'll be able to work with not just that one site, but all other sites that fall under CAcert's umbrella.

Thus a CAcert solution requires less work on the part of the webmaster and it's safer for the users—the latter point having the added advantage of potentially driving more traffic to certain sites, as users who didn't trust the homebrew PKI solution might be more inclined to accept the CAcert trust model instead.

So CAcert is rocking and rolling along, expanding on traditional PKI and offering gads of cool new options for encryption, authentication, digital signing, and the like—and all without robbing its users blind. What's the catch?

Well, there's no catch—just head over to www.cacert.org and check it out for yourself. But there are a few small flies in the ointment.

Fortunately, hackers are well known for jumping into the thick of things and coming to the aid of worthwhile projects…the perfect audience for a subtle call to action. ;)

Root Certificate Inclusion in Browsers

Obviously a major goal for CAcert is to have its root certificate included with all of the popular web browsers, so visitors to secure sites are neither required to download and install the cert themselves nor be subjected to whatever awkward error messages their browser of choice decides to toss at them.

With something like 300 billion people using Windows in southern Florida alone, it's no shock that Internet Explorer is by far the leader when it comes to browser market share. Anecdotal evidence (and common sense) seems to suggest that back during the Browser Wars, commercial certificate authorities probably greased the wheels with a healthy chunk of change to ensure that their root certificates would be included in both Navigator/Communicator and IE—ah, the hidden costs of "strategic partnerships!"

These days, each browser has dramatically different requirements in terms of root certificate inclusion.

In true Microsoft style, Redmond adopted a new metric for determining whether a CA's root certificate is to be included with its browser/operating-system/kitchen-sink product: in order for a CA's root certificate to be accepted—I swear I'm not making this up—said certificate authorities must pay a WebTrust-licensed member of the American Institute of Certified Public Accountants ***up to $250,000*** for an initial evaluation/inspection, plus additional *tens of thousands of dollars* in fees for periodic "follow-ups."(7)

The makers of the Opera web browser did not respond to email queries regarding their inclusion policies/requirements, however a Bermuda-based CA representative stated in the netscape.public.mozilla.crypto newsgroup that "as of [his] last contact in 2003, Opera wanted cash to add a CA [root certificate]. *They currently do not appear to have a standards policy.*"(8,9) Nice to see somebody's got their priorities straight, eh?

Rather than getting into all the other browsers and browser-like programs under the sun, let's jump a bit and discuss open source's favorite son: Mozilla.

Getting in Good with the Lizard

The Open Source advocates among us look forward to a time when software is finally wrenched free from the clutches of its faceless captors—massively proprietary organisations whose interests in innovation seldom reach beyond their own shortsighted marketing strategies, leaving less profitable technologies to stagnate.

And while collaborative software initiatives flourish across the globe, services designed to support and expand the underlying OSS infrastructure continue to face significant challenges. These barriers sometimes arise from corporations leveraging their de facto monopolies against newcomers, but often there's no evil empire to blame. Frequently, bumps in the road are merely the result of various open source advocates and developers disagreeing about one thing or another.

After Netscape disappeared, leaving no one behind to make "executive decisions" about critical things such as root certificate inclusions, the Mozilla Foundation embraced a policy of maintaining the status quo, keeping all existing root certificates installed without really considering what would happen when/if any new CAs came knocking. (10)

(This installed base remained the same even after existing certificate authorities erroneously issued multiple Authenticode certificates labeled "Microsoft Corporation" to a couple of crafty social engineers,(11) arguably demonstrating once and for all that money can't buy you love ***or*** security.)

Trying to go through all the proper channels, developers submitted a "feature enhancment" request to Bugzilla, asking that the CAcert root certificate be included in Mozilla."(7) (This inventive maneuver would pop up in Konqueror's feature request system, also.)(13)

About six months after the Bugzilla request was submitted, an announcement was made inidcating that the CAcert root certificate would be part of the soon-to-be-released Mozilla 1.6.(14)

The announcement momentarily vaulted CAcert's otherwise innocuous request into the public eye—and with all the sudden new exposure came increased scrutiny. While most people were either in favor of the decision or indifferent, some of the more security-minded Mozilla developers voiced concerns.

Despite its nonprofit status, CAcert was criticized for its failure to retain the services of prohibitively expensive third-party auditing firms. As a volunteer-led community certificate authority providing free services to thousands of users, CAcert was in no position to pay for outside consultants.

CAcert is just another two-bit, fly-by-night operation, claimed some of its detractors. There's no oversight, they charged. The whole operation probably just consists of a cable modem, an old Packard Bell laptop, a pirated copy of PC-DOS 3.0, and four lines of Perl code. Their certificates are all encrypted with ROT13. Their private key is stored for safe keeping on a purple Hello Kitty diskette atop Dad's Van de Graaff Generator. Oh, and they spend their free time issuing certificates to serial killers, zombies, and men who bite the heads off kittens. That's right...*kittens*..(15)

Eventually the discussion spilled out of Bugzilla and was was shepherded over to the netscape.public.mozilla.crypto newsgroup. The original Bugzilla feature enhancement request was subsequently blocked/superseded by a directive that the Mozilla Foundation was to develop a formal Certificate Authority acceptance policy before accepting any new root CAs.(16) Wildly disparate proposals for the new acceptance policy flew in from everywhere—people suggested everything from AICPA/WebTrust certification (insanely expensive) to an "open door policy" that would give everybody and anybody who applied access to the root store (insanely reckless)…and every imaginable gradient in between.

I have tremendous respect for all of the individuals who volunteer their time for the Mozilla Foundation, and I can completely understand the fears voiced by those who preferred the status quo. Furthermore, I am certain everyone best intentions at heart...despite the distinct feeling that the discussion had degraded almost to the point of a filibuster.

In some discussions, it seemed as if two or three people were just yelling "NO!" at the top of their lungs without providing any real basis for their concerns—nevertheless, these passionate appeals were frustratingly successful in their ability to steer the debate off-course. I certainly can't fault the individuals involved for trying, of course. For whatever reason, certain people apparently felt that the Mozilla Project was in imminent danger, and so they defended it to the best of their abilities. I have little doubt that I would have done the same, had the roles been reversed.

Fortunately, there is a happy end to this story. After much debate and gnashing of teeth, the CAcert root certificate once again seems on-track for inclusion in the next Mozilla release. (Fingers crossed.)

Looking Ahead

Though the development of a Community-Oriented Certificate Authority doesn't quite reach Kuhn's definition of a true "paradigm shift," it's a revolution nonetheless. Just as when Network Solutions lost its monopoly on domain registration, things have changed significantly for the better. And there's no looking back.

None of us today would consider paying $35 a year to register a top level domain, and very soon VeriSign's $1200+ pricing for SSL certificates will strike us as equally ridiculous—because when you read this article, even if CAcert's root certificate still somehow remains excluded from the basic Mozilla install, the organization will still be growing and gathering momentum. At this point, there's no sense asking if the group will accomplish one thing or another—anything's possible, and it's all just a matter of time.

Says CAcert founder Duane Groth: "[T]he established players in the certificate industry lobby hard to exclude any further competition from entering the market, which means they are able to keep charging exorbitant rates for certificates….This is all set to change.

"Currently there are hundreds of thousands of web browsers out there with our root certificate installed; companies are deploying intranets with certificates issued from CAcert and installing the root certificate on each client machine on the network…. [M]omentum is building at a grass roots level."

Until CAcert's root certificate is preinstalled in your browser of choice, remember that you can always install it manually by visiting www.cacert.org and clicking the appropriate link. And if you're wondering what you can do to help with the effort, join the CAcert mailing list, make suggestions and donations—contribute how you can, if you can. And see the notes at the end of this article for the URLs where you can vote for CAcert's inclusion in Mozilla and Konqueror.

But most importantly: Visit the site, sign up, grab a certificate or two, and start securing your data. Because regardless of what politics may be going on behind the scenes and what seemingly unattainable goals the organisation may set for itself, whether you can spare some time to help with the project isn't the point. CAcert's mission remains the same: to provide you with alternatives to commercial CAs like VeriSign and Thawte, to help you secure your data, and to do the same for the rest of our Internet Community.

It's a crazy world out there, so keep your data safe and your sessions secure. And let us help.

References

1. "The Regulation of Investigational Powers Act (RIPA)," http://www.homeoffice.gov.uk/crimpol / crimreduc/regulation/index.html. 28 Jul 00. See also "U.K. e-mail snooping bill passed," http://www.cnn.com/2000/TECH/computing/07/28/uk.surveillance.idg/: 28 Jul 2000.

2. As of 15 Mar 04, Thawte offered two 128-bit SSL server certificates, priced at $199 and $449 per year, respectively. On that same date, VeriSign offered a host of 128-bit SSL certificate packages ranging from $895 to $1495 per year. (All figures in US$ unless otherwise noted.)

3. CAcert, “Assurance Programme,” http://www.cacert.org/index.php?id=8 (18 Mar 2004).

4. Thawte, “Freemail Web of Trust System,” https://www.thawte.com/cgi/personal/wot/contents.exe (15 Apr 2004). See also Thawte, “thawte: web of trust,” https://www.thawte.com/wot/index.html (18 Apr 2004).

5. William Stallings, “The PGP Web of Trust.” Byte, Feb 1995.

6. Roger Grimes, “Authenticode,” Microsoft TechNet, http://www.microsoft.com/technet/security/topics/secapps/authcode.mspx (18 Mar 04).

7. Microsoft Technet, “Microsoft Root Certificate Program Requirements,” http://www.microsoft.com/technet/security/news/rootcert.mspx (18 Mar 04). See also American Institute of Certified Public Accountants, "WebTrust Program for Certification Authorities: Version 1.0," http://ftp.webtrust.org/webtrust_public/tpafile7-8-03fortheweb.doc: 25 Aug 2000.

8. Emphasis added.

9. Name withheld, “RE: Proposed CA certificate metapolicy,” news://netscape.public.mozilla.crypto: 3 Mar 2004. See also “Re: why and how VeriSign, thawte became a trusted CA?” news://comp.security.misc: 15 Mar 2004.

10. For a list of all the CA root certificates shipped with Mozilla browsers by default, open your copy of Mozilla or Firefox and select Edit -> Preferences -> Privacy & Security -> Certificates -> Manage Certificates -> Authorities.

11. Microsoft Knowledge Base, “How to Recognize Erroneously Issued VeriSign Code-Signing Certificates,” http://support.microsoft.com/default.aspx?scid=kb;en-us;293817&sd=tech (18 Mar 04). See also Microsoft Technet, “Erroneous VeriSign-Issued Digital Certificates Pose Spoofing Hazard,” http://www.microsoft.com/technet/security/bulletin/MS01-017.mspx (18 Mar 04).

12. You too can vote for CAcert root certificate inclusion in the next version of Mozilla. The party’s right here: http://bugzilla.mozilla.org/show_bug.cgi?id=215243.

13. Encourage the KDE Group to include CAcert’s root certificate in the next version of Konqueror. Vote at: http://bugs.kde.org/show_bug.cgi?id=74290.

14. Frank Hecker, “Additional Comment #20,” http://bugzilla.mozilla.org/show_bug.cgi?id=215243: 4 Feb 04.

15. Actually, CAcert is a fully recognized, legally incorporated nonprofit organization with a board of directors, an organizational charter, and a strict set of bylaws that explicitly forbids strategic alliances with zombies or other members of the undead. The CA servers are stored at a secure colocation facility, complete with biometric palm scanners and other cool stuff like that. And nothing is stored or signed in ROT13 format —CAcert has always relied on the far superior Triple-ROT26 algorithm for all cryptography. :)

16. “Mozilla.org needs a public policy on root CA certs,” http://bugzilla.mozilla.org/show_bug.cgi?id=233453 (14 Mar 04).

Cyber*In*security:The Cost of Monopoly (Part 2 of 2)

Authors: Dan Geer, Rebecca Bace, Peter Gutmann, Perry Metzger, Charles P. Pfleeger, John S. Quarterman, Bruce Schnerier

2. MICROSOFT

To sum up this section:

- Microsoft is a near-monopoly controlling the overwhelming majority of systems.
- Microsoft has a high level of user-level lock-in; there are strong disincentives to switching operating systems.
- This inability of consumers to find alternatives to Microsoft products is exacerbated by tight integration between applications and operating systems, and that integration is a long-standing practice.
- Microsoft's operating systems are notable for their incredible complexity and complexity is the first enemy of security.
- The near universal deployment of Microsoft operating systems is highly conducive to cascade failure; these cascades have already been shown to disable critical infrastructure.
- After a threshold of complexity is exceeded, fixing one flaw will tend to create new flaws; Microsoft has crossed that threshold.
- Even non-Microsoft systems can and do suffer when Microsoft systems are infected.
- Security has become a strategic concern at Microsoft but security must not be permitted to become a tool of further monopolization.

Discussion:

Near-monopoly dominance of computing by Microsoft is obvious beyond the findings of any court. That percentage dominance is at peak in the periphery of the computing infrastructure of all industrial societies. According to IDC, Microsoft Windows represented 94 percent of the consumer client software sold in the United States in 2002.[2] Online researcher OneStat.com estimates Microsoft Windows market share exceeds 97 percent.[3] Its Internet Explorer and Office Suite applications share similar control of their respective markets. The tight integration of Microsoft application programs with Microsoft operating system services is a principal driver of that dominance and is at the same time a principal driver of insecurity. The "tight integration" is this: inter-module interfaces so complex, undocumented, and inaccessible as to (1) permit Microsoft to change them at will, and thus to (2) preclude others from using them such as to compete.

Tight integration of applications and operating system achieves user lock-in by way of application lock-in. It works. The absence of published, stable exchange interfaces necessary to enable exchange of data, documents, structures, etc., enlists such data, documents, or structures as enforcers of application lock-in. Add in the "network

effects," such as the need to communicate with others running Microsoft Office, and you dissuade even those who wish to leave from doing so. If everyone else can only use Office then so must you.

Tight integration, whether of applications with operating systems or just applications with each other, violates the core teaching of software engineering, namely that loosely-coupled interfaces make maintenance easier and life-cycle costs lower. Academic and commercial studies supporting this principle are numerous and long-standing. Microsoft well knows this; Microsoft was an early and aggressive promoter of modular programming practices within its own development efforts. What it does, however, is to expressly curtail modular programming and loose-coupling in the interfaces it offers to others. For whatever reason, Microsoft has put aside its otherwise good practices wherever doing so makes individual modules hard to replace. This explains the rancor over Prof. Ed Felten's Internet Explorer removal gadget just as it explains Microsoft's recent decision to embed the IE browser so far into their operating system that they are dropping support for IE on the Macintosh platform. Integration of this sort is about lock-ins through integration too tight to easily reverse buttressed by network effects that effectively discourage even trying to resist.

This integration is not the norm and it is not essential. Just limiting the discussion to the ubiquitous browser, it is clear that Mozilla on Linux or Safari on Macintosh are counter-examples: tight integration has no technical necessity. Apple's use of Safari is particularly interesting because it gets them all the same benefits that Microsoft gets from IE (including component reuse of the HTML rendering widget), but it's just a generic library, easy to replace.[4] The point is that Microsoft has performed additional, unnecessary engineering on their products with the result of making components hard to pull out, and thus raising the barrier to entry for competition. Examples of clean interfaces are much older than Microsoft: the original UNIX was very clean and before that Multics or Dijkstra's 1968 "THE" system showed what could be done. In other words, even when Microsoft was very much smaller and very much easier to change these ideas were known and proven, therefore what we have before us today is not inadvertent, it is on plan.

This tight-integration is a core component of Microsoft's monopoly power. It feeds that power, and its effectiveness is a measure of that power. This integration strategy also creates risk if for no other reason that modules that must interoperate with other modules naturally receive a greater share of security design attention than those that expect to speak only to friends. As proof by demonstration, Microsoft's design-level commitment to identical library structures for clients and servers, running on protocols made explicitly difficult for others to speak (such as Microsoft Exchange), creates insecurity as that is precisely the characteristic raw material of cascade failure: a universal and identical platform asserted to be safe rather than shown in practice to be safe. That Microsoft is a monopoly makes such an outcome the *default* outcome.

The natural strategy for a monopoly is user-level lock-in and Microsoft has adopted this strategy. Even if convenience and automaticity for the low-skill/no-skill user were formally evaluated to be a praiseworthy social benefit, there is no denying the latent costs of that social benefit: lock-in, complexity, and inherent risk.

One must assume that security flaws in Microsoft products are unintentional, that security flaws simply represent a fraction of all quality flaws. On that assumption, the quality control literature yields insight.

The central enemy of reliability is complexity. Complex systems tend to not be entirely understood by anyone. If no one can understand more than a fraction of a complex system, then, no one can predict all the ways that system could be compromised by an attacker. Prevention of insecure operating modes in complex systems is difficult to do well and impossible to do cheaply: The defender has to counter all possible attacks; the attacker only has to find one unblocked means of attack. As complexity grows, it becomes ever more natural to simply assert that a system or a product is secure as it becomes less and less possible to actually provide security in the face of complexity.

Microsoft's corporate drive to maximize an automated, convenient user-level experience is hard to do - some would say un-doable except at the cost of serious internal complexity. That complexity must necessarily peak wherever the ratio of required convenience to available skill peaks, viz., in the massive periphery of the computing infrastructure. Software complexity is difficult to measure but software quality control experts often describe software complexity as proportional to the square of code volume. One need look no further than Microsoft's own figures: On rate of growth, Windows NT code volume rose 35% per year (implying that its complexity rose 80%/year) while Internet Explorer code volume rose 220%/ year (implying that its complexity rose 380%/year). Consensus estimates of accumulated code volume peg Microsoft operating systems at 4-6x competitor systems and hence at 15-35x competitor systems in the complexity-based costs in quality. Microsoft's accumulated code volume and rate of code volume growth are indisputably industry outliers that concentrate complexity in the periphery of the computing infrastructure. Because it is the complexity that drives the creation of security flaws, the default assumption must be that Microsoft's products would have 15-35x as many flaws as the other operating systems.[5]

One cannot expect government regulation to cap code size - such a proposal would deserve the derision Microsoft would heap upon it. But regulators would do well to understand that code "bloat" matters most within modules and that Microsoft's strategy of tight integration makes effective module size grow because those tightly integrated components merge into one. It is likely that if module sizes were compared across the industry that the outlier status of Microsoft's code-size-related security problems would be even more evident than the total code volume figures indicate.

Above some threshold level of code complexity, fixing a known flaw is likely to introduce a new, unknown flaw; therefore the law of diminishing returns eventually rules. The general quality control literature teaches this and it has been the received wisdom in software development for a long time (Lehman & Belady at IBM[6] and later in many papers and at least one book). The tight integration of Microsoft operating systems with Microsoft application products and they with each other comes at a cost of complexity and at a cost in code volume. Patches create new flaws as a regular occurrence thus confirming that Microsoft's interdependent product base is above that critical threshold where repairs create problems. Some end-users understand this, and delay deployment of patches until testing can confirm that the criticality of problems fixed are not eclipsed by the criticality of problems created. With mandatory patches arriving at the rate of one every six days (39 through 16 September), it is few users indeed who can keep up.
Two different subsets of users effectively bow out of the patching game: the incapable - many (end-users who have limited

understanding of - and limited desire to understand - the technology even when it is working correctly) and the critical-infrastructure-few (for whom reliability is such a vital requirement that casual patching is unthinkable). Un-patched lethal flaws thus accumulate in the user community. (The Slammer worm fully demonstrated that point - the problem and the patch were six months old when Slammer hit.)[7] Monopoly market dominance is thus only part of the risk story - market dominance coupled with accumulating exploitable flaw density yields a fuller picture. Not only is nearly every networked computer sufficiently alike to imply that what vulnerability one has, so has another, but the absolute number of known-to-be exploitable vulnerabilities rises over time. Attackers of the most consummate skill already batch together vulnerabilities thus to ensure cascade failure. (The NIMDA virus fully demonstrated that point - it used any of five separate vulnerabilities to propagate itself.)

Microsoft has had a history of shipping software at the earliest conceivable moment. Given their market dominance, within days if not hours the installed base of any released Microsoft software, however ill thought or implemented, was too large to dislodge or ignore. No more. Of late Microsoft has indeed been willing to delay product shipment for security reasons. While it is too early to tell if and when this will actually result in a healthier installed base, it is an admission that the level of security flaw density was a greater threat to the company than the revenue delay from slipping ship dates. It is also an admission that Microsoft holds monopoly power - they and they alone no longer need to ship on time. That this coincides with Microsoft's recent attempts to switch to annual support contracts to smooth out their revenue streams is, at least, opportunistic if not tactical.

On the horizon, we see the co-called Trusted Computing Platform Association (TCPA)[8] and the "Palladium" or "NGSCB" architecture for "trusted computing." In the long term, the allure of trusted computing can hardly be underestimated and there can be no more critical duty of government and governments than to ensure that a spread of trusted computers does not blithely create yet more opportunities for lock-in. Given Microsoft's tendencies, however, one can foresee a Trusted Outlook that will refuse to talk to anything but a Trusted Exchange Server, with (Palladium's) strong cryptographic mechanisms for enforcement of that limitation. There can be no greater user-level lock-in than that, and it will cover both local applications and distributed applications, and all in the name of keeping the user safe from viruses and junk. In other words, security will be the claimed goal of mechanisms that will achieve unprecedented user-level lock-in. This verifies the relevance of evaluating the effect of user-level lock-in on security.

3. IMPACT ON PUBLIC PROTECTION

To sum up this section:

- Without change, Microsoft's history predicts its future.
- We must take conscious steps to counter the security threat of Microsoft's monopoly dominance of computing.
- Unless Microsoft's applications and interfaces are available on non-Microsoft platforms it will be impossible to defeat user lock-in.
- Governments by their own example must ensure that nothing they deem important is dependent on a monoculture of IT platforms; the further up the tree you get the more this dictum must be observed.
- Competition policy is tangled with security policy from this point on.

Discussion:

Microsoft and regulators come to this point with a considerable history of flouted regulation behind them, a history which seems unnecessary to recount other than to stipulate that it either bears on the solution or history will repeat itself.

Yes, Microsoft has the power to introduce features unilaterally and one might even say that the current security situation is sufficiently dire that Microsoft as the head of a command structure is therefore somehow desirable. Yet were it not for Microsoft's commanding position economics would certainly be different whether it would be a rise in independent, competitive, mainstream software development industries (because the barriers to entry would be lower), or that today's locked-in Microsoft users would no longer pay prices that only a monopoly can extract. For many organizations the only thing keeping them with Microsoft in the front office is Office. If Microsoft was forced to support Office on, say, Linux, then organizations would save substantial monies better spent on innovation. If Microsoft were forced to interoperate, innovators and innovation could not be locked-out because users could not be locked-in.

Both short-term impact mitigation and long term competition policy must recognize this analysis. In the short term, governments must decide in unambiguous ways whether they are able to meaningfully modify the strategies and tactics of Microsoft's already-in-place monopoly.

If governments do not dismantle the monopoly but choose instead to modify the practices of the monopoly they must concede that that route will, like freedom, require eternal vigilance. Appropriate support for addressing the security-related pathologies of monopoly would doubtless include the introduction of effective, accessible rights of action in a court of law wherever security flaws lead to harm to the end-user. In extreme cases, the consequences of poor security may be broad, diffuse, and directly constitute an imposition of costs on the user community due to the unfitness of the product. Under those circumstances, such failures should surely be deemed "per se" offenses upon their first appearance on the network.

Where risk cannot be mitigated it can be transferred via insurance and similar contracts. As demonstrated in previous sections, the accumulation of risk in critical infrastructure and in government is growing faster than linear, i.e., faster than mere counts of computers or networks. As such, any mandated risk transfer must also grow faster than linear whether those risk transfer payments are a priori, such as for bonding and insurance, or a posteriori, such as for penalties. If risk transfer payments are to be risk sensitive, the price and probability of failure are what matter and thus monopoly status is centrally relevant. For governments and other critical infrastructures, the price of failure determines the size of the risk transfer. Where a software monoculture exists in other words, a computing environment made up of Windows and almost nothing else what remains operational in the event of wholesale failure of that monoculture determines the size of the risk transfer. Where that monoculture is maintained and enforced by lock-in, as it is with Windows today, responsibility for failure lies with the entity doing the locking-in in other words, with Microsoft. It is important that this cost be made clear now, rather than waiting until after a catastrophe.

The idea of breaking Microsoft into an operating system company and an applications company is of little value - one would just

inherit two monopolies rather than one and the monocultural, locked-in nature of the user base would still nourish risk. Instead, Microsoft should be required to support a long list of applications (Microsoft Office, Internet Explorer, plus their server applications and development tools) on a long list of platforms. Microsoft should either be forbidden to release Office for any one platform, like Windows, until it releases Linux and Mac OS X versions of the same tools that are widely considered to have feature parity, compatibility, and so forth. Alternately, Microsoft should be required to document and standardize its Exchange protocols, among other APIs, such that alternatives to its applications could independently exist. Better still, split Microsoft Office into its components - noticing that each release of Office adds new things to the "bundle": first Access, the Outlook, then Publisher. Even utilities, such as the grammar checker or clip art manager, might pose less risk of compromise and subsequent OS compromise if their interfaces were open (and subject to public scrutiny and analysis and validation). Note that one of the earlier buffer overflow exploits involved the phone dialer program, and ordinarily benign and uninteresting utility that could have been embedded within dial-up networking, Internet Explorer, Outlook and any other program that offered an Internet link.

The rigorous, independent evaluations to which these otherwise tightly integrated interfaces would thus be exposed would go a long way towards security hardening them while permitting meaningful competition to arise. Microsoft will doubtless counter that its ability to "innovate" would be thus compromised, but in the big picture sense everyone else would have a room to innovate that they cannot now enjoy.

Where governments conclude that they are unable to meaningfully modify the strategies and tactics of the already-in-place Microsoft monopoly, they must declare a market failure and take steps to enforce, by regulation and by their own example, risk diversification within those computing plants whose work product they value. Specifically, governments must not permit critical or infrastructural sectors of their economies to implement the monoculture path, and that includes government's own use of computing. Governments, and perhaps only governments, are in leadership positions to affect how infrastructures develop. By enforcing diversity of platform to thereby blunt the monoculture risk, governments will reap a side benefit of increased market reliance on interoperability, which is the only foundation for effective incremental competition and the only weapon against end-user lock-in. A requirement that no operating system be more than 50% of the installed based in a critical industry or in a government would moot monoculture risk. Other branches to the risk diversification tree can be foliated to a considerable degree, but the trunk of that tree on which they hang is a total prohibition of monoculture coupled to a requirement of standards-based interoperability.

CODA

These comments are specific to Microsoft, but would apply to any entity with similar dominance under current circumstances. Indeed, similar moments of truth have occurred, though for different reasons, with IBM or AT&T. The focus on Microsoft is simply that the clear and present danger can be ignored no longer.

While appropriate remedies require significant debate, these three alone would engender substantial, lasting improvement if Microsoft were vigorously forced to:
" Publish interface specifications to major functional components of its code, both Windows and Office.
" Foster development of alternative sources of functionality through an approach comparable to the highly successful 'plug and play' technology for hardware components.
" Work with consortia of hardware and software vendors to define specifications and interfaces for future developments, in a way similar to the Internet Society's RFC process to define new protocols for the Internet.

BIOGRAPHICAL INFORMATION

Daniel Geer, Sc.D - Dr. Geer is Chief Technical Officer of @Stake, in Cambridge, Mass. Dr. Geer has a long history in network security and distributed computing management as an entrepreneur, author, scientist, consultant, teacher, and architect. He has provided high-level strategy in all manners of digital security and on promising areas of security research to industry leaders including Digital Equipment Corporation, OpenVision Technologies, Open Market, and CertCo. He has written extensively on large-scale security issues such as risk management, applications of cryptography, and Web security for The Digital Commerce Society, the Securities Industry Middleware Council, the Internet Security Conference, and the USENIX Association for whom he founded several conferences.

Dr. Geer has testified before Congress on multiple occasions and has served on various relevant advisory committees to the Federal Trade Commission, the National Science Foundation, the National Research Council, the Commonwealth of Massachusetts, the Department of Defense, the National Institute of Justice, and the Institute for Information Infrastructure Protection.

Dr. Geer holds several security patents, an Sc.D. in Biostatistics from Harvard University's School of Public Health and an S.B. in Electrical Engineering and Computer Science from MIT.

Charles P. Pfleeger, Ph.D - Dr. Pfleeger is a Master Security Architect in the Professional Services group of Exodus Communications, Inc. From 1992 to 1995 he was Director of European Operations for Trusted Information Systems, Inc. (TIS) and head of its European office in London. He was a member of the author group of the U.S. Federal security evaluation criteria and a co-author of the evaluation criteria for trusted virtual machine architectures. He led activities in secure networking, security analysis in hardware design, secure system architecture, and research into assured service. Prior to joining TIS in 1988, he was a professor in the Computer Science Department of the University of Tennessee Dr. Pfleeger has lectured throughout the world and published numerous papers and books. His book *Security in Computing* (the third edition will be available from Prentice Hall in 2002) is the standard college textbook in computer security. He is the author of other books and articles on technical computer security and computer science topics.

He holds a Ph.D. degree in computer science from The Pennsylvania State University and a B.A. with honors in mathematics from Ohio Wesleyan University.

Bruce Schneier - Internationally renowned security expert Bruce Schneier has authored six books --including BEYOND FEAR and SECRETS AND LIES-- as well as the Blowfish and Twofish encryption algorithms. Mr. Schneier has appeared on numerous television and radio programs, has testified before Congress, and is a frequent writer and lecturer on issues surrounding security and privacy.

Mr. Schneier is responsible for maintaining Counterpane's technical

lead in world-class information security technology and its practical and effective implementation. Mr. Schneier's security experience makes him uniquely qualified to shape the direction of the company's research endeavors, as well as to act as a spokesperson to the business community on security issues and solutions.

Mr. Schneier holds an MS degree in computer science from American University and a BS degree in physics from the University of Rochester.

John S. Quarterman - John S. Quarterman is founder of InternetPerils, an Internet risk-management company. Previously, he was Founder and Chief Technology Officer of Matrix NetSystems Inc., the first company to map and track global traffic across the Internet. Mr. Quarterman has almost thirty years experience with network issues dating as far back as 1974, when he first used ARPANET, the Internet's predecessor, at Harvard University. He subsequently worked on ARPANET Unix software for Bolt, Beranek and Newman, the original prime contractor for the network.

Mr. Quarterman has consulted for a wide range of companies and organizations, including AT&T, HP, IBM, MCI and Nortel, among others. Twice elected to the board of directors of USENIX, he was instrumental in the board's decision to provide funding for UUNet, one of the first two commercial Internet service providers. A published author, he has written for Communications of the ACM, Forbes, First Monday and Computerworld, among others. He has appeared in articles written by others in the New York Times, the San Jose Mercury News, The Economist, The Washington Post, Wired and others too numerous to mention.

Perry Metzger - Perry Metzger is managing partner of Metzger, Dowdeswell & Co LLC, a New York based computer security and infrastructure consulting firm. Prior to this, Mr. Metzger founded and served as CEO of Wasabi Systems, Inc., a startup specializing in operating system software for embedded platforms. Previously Mr. Metzger served as President of Piermont Information Systems Inc., a New York based computer security consulting firm he founded in 1994. Piermont's clients included prominent international banks and brokerages, money management companies, public relations firms and advertising agencies

Before founding Piermont, Mr. Metzger was involved in a variety of innovative technological projects, including highly parallel computer systems, automated equities trading systems, automated systems management software, and the implementation of one of the world's first firewall systems. Mr. Metzger is highly active in the work of the Internet's standardization body, the IETF. He was instrumental in the design and standardization of several major internet security protocols,including IPSEC, for which he served as co-author of several of the initial standards documents.

Becky Bace - Becky Bace is an internationally recognized expert in network security and intrusion detection. A 2003 recipient of Information Security Magazine's Women of Vision Award, she is recognized as one of the most influential women in Information Security today. Ms. Bace has worked in security since the 1980s, leading the first major intrusion detection research program at the National Security Agency, where she received the Distinguished Leadership Award, serving as the Deputy Security Officer for the Computing Division of the Los Alamos National Laboratory, and, since 1997, working as a strategic consultant.

She is currently President and CEO of Infidel, Inc., a security consulting firm. Ms. Bace's publication credits include the books *Intrusion Detection* (Macmillan, 2000) and *A Guide to Forensic Testimony: The Art and Practice of Presenting Testimony as An Expert Technical Witness*, (Addison-Wesley, October, 2002).

She received a B.S., Engineering/Computer Science from the University of the State of New York, and an M.E.S., Digital Systems Engineering, from Loyola College.

Peter Gutmann - Peter Gutmann is a researcher in the Department of Computer Science at the University of Auckland working on design and analysis of cryptographic security architectures. He helped write the popular PGP encryption package and has authored a number of papers on security and encryption including the X.509 Style Guide for certificates.

Over the years, Mr. Gutmann has uncovered numerous security flaws in various computing products, including problems with the encryption used in an early version of the Netscape browser and, later, Internet Explorer. He has also uncovered flaws in previous versions of Norton's Diskreet disk encryption, the Windows 95 password file system and the smart-card fare system used by Auckland's largest public transportation organization. Gutmann is the author of the much used, open source cryptlib security toolkit.

---

[2] "Wal-Mart sells more Linux wares online," Matt Hines, News.com. August 21, 2003.
[3] "Microsoft's Windows OS global market share is more than 97% according to OneStat.com," OneStat.com press release. September 10, 2002.
[4] "Apple Releases its own browser," Joe Wilcox, News.com, January 7, 2003.
[5] Microsoft seems at least aware of the problem. See: http://www.wired.com/wired/archive/3.09/myhrvold.html.
[6] L.A. Belady and M.M. Lehman, "A Model of Large Program Development," IBM Systems Journal 15(3), p.225 252 (1976).
[7] "Slammer worm brings patch mgmt. issues to the fore," Audrey Rasmussen, Network World Fusion, Feb. 5, 2003.
[8] See: http://www.trustedcomputing.org/home

This article is re-printed with permission. The originals can be found at:

http://www.ccianet.org/papers/cyberinsecurity.pdf

Book review

Author: Greg Lehey <Greg.Lehey@auug.org.au>

PRACTICAL VOIP

by Luan Dan, Cullen Jennings and David Kelly
O'Reilly and Associates, 2002

It's easy to see that “Practical VoIP” is a book aimed at people in the know. You have to look at the fine print on the back cover to discover that VoIP means Voice over IP, and even then the average browser at Dymocks is liable to be none the wiser. Unless you're really interested in using the Internet for telephone connections, you're liable to pass over this book.

Looking back at the front cover, you're bombarded with buzzwords: the subtitle (which O'Reilly writes at the top of the cover) states

"MGCP, H.323, SIP, RTP, COPS, RADIUS, and More". In all probability, even the experienced reader won't know more than one or two of these terms.

More to the point, though, is the rest of the title: the full title is "Practical VoIP using VOCAL". That's a pretty accurate title: it's basically a VOCAL user manual. The first of the 19 chapters explains what VOCAL is: it's free VoIP software supplied by Vovida Networks, a Silicon Valley startup associated with Cisco. The authors all work for Cisco.

The introductory chapter also gives an overview over the VOCAL architecture. In the process, it also introduces many VoIP concepts, with copious references to the RFCs on which VOCAL is based. For the beginner, this is possibly the most important part of the book.

Chapter 2 describes how to set up VOCAL at home, somewhat hampered by the superficial treatment of those components which are not part of VOCAL, such as telephones. It also (finally) tells you the requirements for running VOCAL: it runs under Linux and probably other UNIX variants. For AUUG members, that's not an issue, but it could prove a bit confusing for the average browser, assuming he hasn't been put off by the buzzwords on the cover. We discover that the phones can be (comparatively expensive) IP phones, or also "softphones", based on a PC, a sound card and a headset maybe. The softphone described in the chapter has a few restrictions: in particular, it does not support voice communications. It is sufficient for checking the VOCAL functionality and configuration, however. To quote: "Once you get past this stage and start making real calls with a phone or sound card, you will experience some of the excitement that we've experienced at Vovida".

Finding softphones is not easy. The book provide URLs where you can get softphones; unfortunately, none of them refer to production-quality implementations. During my review I was not able to find a usable softphone, though I'm told that they exist.

This is clearly not a book for somebody dabbling in VoIP. Before buying the book, you'll already have had to make a commitment, including specific hardware not described in detail in this book.

Once you've made your decision on your VoIP setup, and assuming you have decided that VOCAL is for you, this book is a very helpful user guide. It goes beyond the traditional installation guide and explains how VOCAL fits into the overall structure. Chapter 3 describes how to set up a larger trial system, and the remaining chapters describe individual topics, including provisioning users and servers, base system configuration, interaction with other systems, including MGCP gateways and H.323 endpoints. Interspersed between these chapters are details of the protocols and languages involved, including SIP, SDP, TRIP and an overview of how the VOCAL code is structured.

In summary, this book appears to be an excellent second book on VoIP, following on from "An introduction to VoIP". Unfortunately, I haven't been able to find this book. At the time of going to press, the editor at O'Reilly had not answered my query about whether O'Reilly is planning something in this area.

Overheard in the Office

Author: Anonymous

[Editor's note: The following piece was sent in to your editor and and the nod was given to it being published, as long as the CIO who submitted it remained anonymous, for his employer's piece of mind]

Just got The Call™ from the professional anti-Linux SWAT team at Microsoft Australia. Chewed through an hour of their time, and left them a little....speechless, I think is a good term. Certainly they were on defence for nearly the whole hour (and not doing terribly well at it, I might add). I don't think they were at all equipped for the kind of approach that I took.

I certainly didn't bag their products. Not entirely. I actually tried to be helpful ("The stuff you make has _so_ much _potential_, it's a pity that your company philosophy seems to be focused more on rendering the products non-interoperable than in taking advantage of that.")

There was a team of two, on a speaker-phone. I got the impression that there was a third person who didn't speak.

I pointed out that we (myself, management, and other IT professionals and managers with whom I come into contact) just don't _trust_ Microsoft. They asked why. I quoted some choice pieces of Bill's testimony in Microsoft vs DoJ, and some of his Office 2003 launch speech (If you ever deal with these guys, have a web-browser handy), plus the CSS1 patent issue, Microsoft vs the EU, and Microsoft's commitment to security and engineering.

'Well, of course,' they said, 'up until now Microsoft's been very Marketing driven, and has often released products to marketing deadlines before they were ready. But that's all changed. Engineering rules the releases now. They release when they're ready and Marketing can't change that.'

I laughed.

'Perhaps you should have applied that to Windows 2003 Server. Have you _used_ it? I mean _seriously_?'

They hadn't, of course. They haven't used the products they're trying to convince _me_ to use.

'Besides, that's what the Microsoft Rep told me _last_ year. And the year before that. And the year before that. Security is number one. We're totally committed to quality products. Everything was rubbish before but it's all better now. Maybe it's even true this time, but you've cried wolf once too often. Perhaps when Longhorn's been out and about for a year, we can see if those words are backed up, but those dodgy little point-releases (I'm talking XP and 2003 server here) are not really helping _you_ demonstrate to _me_ that this time is any different to the last dozen times I've been told all this.'

Everything got very quiet at this point. So I jumped in again.

'You know what would really help you folks? Interoperability. I mean, I can go get me a Sun box, a Linux box, a BSD box, an AIX box, a Mac OSX box, and I can get them to all play nice. They all play together. There's only one odd man out. Weird cousin Bill who won't come to the party unless nobody else turns up. It's just easier not to invite him, right? Why take an open, published standard and then make a trivial, unpublished non-interoperable change?'

'Yes, we do deliberately make changes to prevent interoperability', said one. I thought that was a very brave admission, then, 'but we make those changes to provide direct business benefits.'

'Umm. Whose business, exactly? What direct business benefit do I get from a non-interoperable Kerberos implementation? What business benefit do I get from abbreviated single-synch TCP handshakes? And how much do those benefits cost me?'

Again, the team floundered, and tried to change the subject.

'You know we're only doing this because we believe in the products.'

'Absolutely,' I assured them, 'I have no doubt at all that you believe in and support your company's products. After all, it would take a monstrous hypocrite to sit there and do this just for the money, right?'

Some nervous laughter.

'Look, ' I said, 'It's great that you want to put all these nifty, friendly cuddly end-user features into your products, but many of those really don't have much place on the business desktop. Thankfully, you gave us group policies, where we can turn nearly all of that hard and expensive effort off. And we do. Ask most of the sysadmins at a Microsoft only shop.'

Then I got an invitation to attend some little CIO get-togethers. I hrmed a bit. "Like you'd pay for my time?"

'Oh, absolutely.', I was quickly assured.

History of the transport of computer viruses via email

Author: Edwin Groothuis <edwin@mavetju.org >

Remember the old rule of the thumb regarding email and viruses? "As long as you don't run the attachment, you are safe."

The computer world has evolved since then...

History

Up to the release of MS-Word 6, the line between safe and unsafe attachments in your email was simple: if the file was executable (i.e. did it end with .exe, .com or .bat), then it was not safe. Was it is a text file, a document, a spreadsheet, then it was safe to open.

Breaching the line between data and executables

MS-Word 6 introduced a new feature: A scripting language in the word-processor, and it was installed with scripting enabled by default. There were a couple of issues with it:

- The file with executable code called NORMAL.DOT, which will be executed when Word was started.
- The introduction of AutoMacros, executable code in the documents which is ran when the document is being opened.
- The ability of the executable code to update the file NORMAL.DOT.

The thin border between safe and unsafe attachments was breached: documents could suddenly have executable payload.

Concept virus

The first Word virus was the Concept Word Macro virus. It didn't do much besides displaying a dialogbox with a "1" on the screen when an infected document was loaded and infecting other documents with itself. It didn't do any damage further, it was just annoying.

Mixing Word and Outlook

The next feature was the capability of the Word scripting language to interact with the MS Outlook mail reader. Where a Word viruses up to that point was only able to propagate slowly via shared Word documents, it suddenly had access to a faster path: It could send itself via Outlook to everybody in the address book on the infected computer, and with a little bit of luck the recipients would open the Word document and the infection would spread.

Beside the technical capabilities to get this virus spreading, the virus needed to convince the receiver that it was safe to open the attachment. Welcome to the social engineering department. The first step is to make sure the receiver trusts the source. Since the virus gets the receivers' address from the address book of the user whose Word document is infected, that means there is some kind of trust relationship between the sender and receiver, and thus most likely also between the receiver and the sender. The second step is to use a catchy text in the body while referring to the attachment, for example by referring to the document as a shared secret between the sender and receiver.

Melissa virus

The first virus which successfully combined these two features was the Melissa virus. When a user opens an infected document, the virus is activated and it infects the NORMAL.DOT file so all newly created documents will contain the virus; sends itself in an email with the following text to the first 50 people in the adddress book of the infected user:

> Subject: Important Message From username
> Here is that document you asked for ... don't show anyone else ;-)'

It is hard to resist emails with this text, even if you are made suspicious by the fact that you got five of them already, all from different sources...

Get rid of the users

When displaying emails, most of the modern browsers only show the text and/or the HTML parts of the email. The last line of defense with regard to email based viruses was the fact that users needed to open the attachment before the virus became active.

If you could execute code in the HTML part of a document, you would be able to infect the computer without having to open an attachment.

Fortunately this is not so easy anymore. JavaScript, one of the programming languages which can be embedded in HTML, isn't capable of accessing files on the local disk. Java, a programming language which is often embedded in web based applications, doesn't allow remote applications to access local disks. ActiveX, Microsofts competitor of Java, shouldn't allow remote applications to access local disks, but due to some implementation issues this was sometimes still possible.

So if you were able to make an HTML message with ActiveX components which bypassed the ActiveX security, you would be able to create a file on disk from just viewing the email message.

BubbleBoy virus

The BubbleBoy virus was the first virus which contained ActiveX code which bypassed the security and wrote a file to disk to the startup directory of Windows. That was all the email did. But when the computer was restarted, the file was started and the the virus started to propagate to everybody in the address book of the infected user.

This is not an executable

After these viruses, the users knew they had to keep an eye open for attachments which were executables, and Word and other MS-Office related documents. Luckily, images and audio files (GIFs and MP3s for example) are still safe. So a quick visual inspection of the name of the attachment should tell if it is safe or not.

MS-Outlook, for example, doesn't by default display the full filename. So an attachment with the name 'test.doc' would be displayed as a 'test' with a Word document icon above it. And an attachment with the name 'test.gif.exe' would be displayed as 'test.gif' with an executable icon above it. If the user only checks the filename displayed, he would see the image filename and assume it was safe to open. Of course he would know immediately know he had ran a program instead of having opened an image, but the damage is done.

Badtrans virus

The payload of the Badtrans virus appeared under the filenames of README.TXT.exe and s3msong.MP3.pif. When the extension wasn't shown, the filenames looked innocent. When opening the attachment the executable was run. To soothe the user into thinking that the application hadn't ran, it showed a dialog box saying "File data corrupt: probably due to bad data transmission or bad disk access".

Faster transmission, obscuring the sender and guessing the recipients

---

By sending infected emails through the mail application on the infected computer, MS-Outlook creates some side effects:

- The email has the sender-address of the user, so tracking of the infected computer is easy.
- The email is delivered to the SMTP gateway of the ISP which can scan it for viruses and thus can block it.
- The SMTP gateway of the ISP might block email which doesn't have an internal sender address.
- The outgoing mail queue on the SMTP gateway of the ISP will grow and the CPU usage on the virus scanner on the SMTP gateway will rise. If properly monitored, alerts will go out to the ISP.
- The SMTP gateway of the ISP will get records of undelivered emails and it might alert them.

To overcome these problems, viruses started to be designed with their own SMTP engines. This way the ISP would be circumvented and would it be harder for the sender to find out where the infected emails were coming from.

Because the sender address could be faked now, more social engineering tricks can be performed. For example, email can be faked to come from the MAILER-DAEMON, which is normally computer generated email coming from SMTP gateways informing you that the email sent couldn't be delivered, and often attaching the ful original email to the bounced message. Opening the attachment is one of the ways to find out which email wasn't able to be delivered.

To counter these kind of viruses, ISPs started to block all outgoing SMTP sessions except the ones coming from and to their own SMTP gateways.

Frethem virus

The Frethem virus was the first virus with its own SMTP engine on board.

MyDoom virus

The MyDoom virus was one of the first viruses with its own SMTP engine on board and which also sends emails to common names (bill, john, mike etc) of target domains. The faked sender addresses will get the undeliverable messages.

Blocking the virus scanner

The virus scanner on both the SMTP gateway of the ISP and the computer which retrieves the email should be able to open the attachments to check for viruses. But what if the attachment is an encrypted ZIP file and the key to open it is given in the email? Unfortunately, there is no clear method to prevent problems with these kind of email viruses.

Mimail

The Mimail-M virus had an encrypted ZIP archive attached:

For unzip archiver download WinZip:
http://download.winzip.com/winzip81.exe
Password for archive is "kiss".
Attached file: wendy.zip

Relevant links

- Word Macro Viruses by Bruce P. Burrell - bpb@umich.edu: http://www.itd.umich.edu/virusbusters/hist-word-macro-

viruses.html

- Virus information about the Concept Word Macro virus: http://www.sophos.com/virusinfo/analyses/wmconcept.html
- Virus information about the Melissa Word Macro virus: http://www.sophos.com/virusinfo/analyses/wm97melissfam.html
- BubbleBoy Virus Changes The Rules: http://www.netlawtools.com/security/bubbleboy.html
- VBS/BubbleBoy Analysis by Ian Whalley, IBM Research, USA: http://www.virusinfo.bz/VB/VbsBubbleboy.htm
- W32.Badtrans.gen@mm: http://securityresponse.symantec.com/avcenter/venc/data/w32.badtrans.gen@mm.html
- Self-Propagating Worm Roaming Internet: http://www.esecurityplanet.com/trends/article.php/10751_1367621
- W32/Frethem-E: http://www.sophos.com/virusinfo/analyses/w32fretheme.html
- W32/MyDoom-A: http://www.sophos.com/virusinfo/analyses/w32mydooma.html
- W32/Mimail-M: http://www.sophos.com/virusinfo/analyses/w32mimailm.html

Book reviews

Author: Michael Still <mikal@stillhq.com>

Just for Fun: The story of an accidental revolutionary by Linus Torvalds and David Diamond. Texere 2001 (ISBN 1-58799-080-6)

Just for fun, the story of an accidental revolutionary, is a jolly good read. It's conversation style, and lighthearted attitude make it accessible, and quite compelling. This book would be a good introduction into the motivations behind some people's contributions to open source for the uninitiated.

Just for Fun is the story of Linus' life up to about 2001, told with the assistance of David Diamond (a journalist). The age of the book means that some of the book reads as a little out of date, for example Linus speaks extensively about the dot com bubble, which obviously hadn't burst at the time that Linus is writing.

Rather conveniently, it appears that Linus has had a very easy to read life, although perhaps it is the literary style. The book is interesting, fun, and very very easy to read. Importantly to me, Linus doesn't pretend to have written the entire book himself, which is clearly not the case. Interspersed throughout the book are commentary segments written by David Diamond. Linus also seems to have his head screwed on the right way — the irony of his position as a geek god by accident doesn't escape him. There are even passages in the book where you can see glimpses of Linus himself questioning his own fame.

Just for Fun starts and ends with Linus attempting to explain his theory of the meaning of life, which is something along the lines of that all things go through three evolutionary stages: survival; social order; and the entertainment. For example, people started fighting initially to ensure survival, then they fought for social status, and now wars are waged as a form of entertainment on CNN and gaming consoles everywhere. Linus argues that a similar progression applies to software as well.

The book discusses technical issues briefly, but is written in a way which is accessible for non-technical people. Linus and David spend a fair bit of time trying to explain open source, and what motivates people to contribute to open source. Just for fun would be a good book to hand to someone who wants to understand open source more, but isn't openly hostile to the concept initially.

To me, as someone already comfortable with open source, the best bit of the book was starting to understand what motivated Linus to start writing Linux. I've been fielding lots of questions about that at work recently, and now I have some sort of answer that actually makes sense forming in the back of my brain. In summary, Linus was a geek who hid in his room a lot — not just because of the fantastic weather Finnish winter weather, his rather traumatic sounding childhood had something to do with it as well — who wanted to learn how to use his new 386 better. He therefore wrote a terminal emulator. Next, he wanted to be able to save files to disc in his terminal emulator, which ran on the bare 386 hardware, so he added a disc driver. After this, it dawned on Linus that he had started an operating system.

Despite it's age, Just for fun is a jolly good read, and I would recommend it.

Book reviews

Author: Doug Jackson <Doug.Jackson@citadel.com.au>

The Complete FreeBSD

In the early 1990's, I set up a series of 'Terminal Servers' at my workplace to allow networking staff to control a set of Cabletron Token Ring hubs located in remote offices, using the wonderful terminal program ‘tip’, and a serial link. From memory, we were using Zenith 80386 systems, with 40Mb hard disks. With careful installation, you could install FreeBSD 1.5 onto the systems, and have a really useful system.

Hasn't the world changed? What for me started as a stack of floppy disks labeled 386/BSD-0.9 has turned into a useful workhorse, albeit over a significant number of revisions. I have been advocating FreeBSD at various workplaces, even using it to introduce 'Samba' for file sharing, so I eagerly awaited the opportunity to examine Greg Lehey's book called "The Complete FreeBSD."

The first thing that I saw was the using the installer to slice up the disks is no longer the 'accepted way' of doing things... Lehey recommends a / file system of 4Gb, not having a separate /usr or / var file system, and allocating the rest of the disk to a /home file system.

Having actively avoided the various 'Linux' distributions for more than the last 10 years, I was very pleased to see that Lehey provides all of the missing installation and running information that I have at various stages learnt, and subsequently forgotten. Finally all of the information is in one place.

I found that the book clearly explained all of the setup processes to fully configure a system, including DNS, routing, getting the Web server running, SSH, and setting up NFS. Everything is extremely clear, with numerous examples of detailing configuration settings etc.

The chapter that details setting up an X server was truly 'gem' like. I have literally spent hours hand crafting XF86Config files to use my monitor at bizarre resolutions, and hours debugging it when I stuffed it up. Lehey's explanation was clear and concise, and covered things that I was not aware could be done, like configuring multiple monitors on a single system, or using multiple X servers on a single system (Of course, now that I think about it, of course it should be able to be done, but the first step is to know about it…).

An area that I have never fully understood is how the system boots, and what the loader could be used for. The book was very helpful in explaining the boot process, and how the loader can be used for diagnostic work. On top of this, the book explains the 'PXE' boot process and how I can configure a diskless system to network boot, something that I through could only be done with one of my Sparc stations.

An area that is extremely important is maintaining currency with new releases. As exploits are discovered for software, and fixes are made, it is important to keep your system up to date. The book has a complete chapter that is devoted to how to use CVS to ensure that your system is running the latest code.

My only criticism is that the book lacks information on how to network FreeBSD systems and Macs. It covers Samba clearly, but ignores Macs. Once upon a time, you could use 'CAP', but it looks like that is long discarded. I understand that you can use netatalk, which is in the BSD kernel, but it seems like it needs a seed router. It would have been nice to see this sort of specialised information included in the book.

In all, the book should probably be re-titled to be 'BSD systems for people who are busy, and don't have time to look things up." I thoroughly recommend it to anybody who is interested in BSD systems.

FreeBSD 5.2 Review

Author: Jem Matzan

The License

You'd be hard pressed to find a license less restrictive than the BSD License. Basically it says two things: that anyone redistributing the software must include all copyright notices and the appropriate license agreements, and that the FreeBSD Project and all contributors to the project may not be held liable for the software if problems should arise with it. It places no limitations on what you can do with the code; that means that it's Free Software. You can make unlimited copies of it, install it on any number of machines, give it to all of your friends and family, modify it in any way that you see fit, and even sell it if you want.

Most people are more familiar with the GNU GPL. The primary difference between the GNU General Public License and the BSD License are the restrictions that the GPL imposes on publishers. Where the BSD License allows anyone to take BSD code and turn it into something proprietary and closed-source, the GPL does not. This may seem "more free" than the GPL, but it doesn't protect the rights of end-users the way the GPL does because it does not require the publisher to make the source code available. In this sense the BSD License is seen as friendlier to commercial software companies (and hardware companies that use embedded software); in fact Microsoft at one point took a great deal of BSD code relating to networking to include in early versions of Windows NT.

Overview

There are two main editions of FreeBSD: the development edition (which includes STABLE and CURRENT), and the more stable version with more mature code, called RELEASE. There are two parts to RELEASE: the new technology release (which is, as of this writing, at version 5.2) and the production release (which is, as of this writing, at 4.9). The two are very different, with the former being a tested CURRENT snapshot and the latter being a product of extensive testing over a longer period of time. If you have newer hardware (made within the past 18 months) and you're interested in using FreeBSD for experimentation or for regular desktop use, 5.2-RELEASE is likely to be your best choice because of its expanded hardware support. The development team strongly cautions 5.2 users that there could be possibly damaging bugs in the 5.2-CURRENT code, and they're not kidding -- my machine won't even boot a CURRENT build (as of 1/16/04) because of problems with the way ATA drives are handled.

CURRENT is for developers and those who wish to beta test in-progress code, much like using the development or beta version of any other operating system or software package. It may contain debugging features that slow things down, and it may contain broken code that makes your system unbootable or causes data loss. The RELEASE edition is determined by a roadmap designed by the Release Engineering Team; once all of the goals for the next release are accomplished, the code is frozen (in terms of new additions, not bug and security patches) and release candidates are made to perform wider-scale testing. Once the known release-critical bugs are squashed, RELEASE is released.

STABLE is for the production release and it is not what its name implies. The STABLE edition is comprised of mature CURRENT code that is being tested and considered for the next production release. At this point the production release is several versions behind the new technology release because the 5.x code simply isn't old enough yet (meaning mature and tested thoroughly with all planned features fully implemented).

The source tree (kernel + userland) is developed as one functioning unit, not as scattered projects as in the GNU/Linux world. This makes for greater system reliability and cleaner code. Beyond that is the Ports tree, which includes the program source code plus any FreeBSD-specific patches for over ten thousand ported applications. If a GNU/Linux program isn't ported to FreeBSD it can still be used if you have binary emulation enabled, and with regard to that there are hundreds of GNU/Linux programs in the Ports tree that can run flawlessly by using the Linux binary compatibility module.

Clarifications

There are two common misconceptions about FreeBSD that I would like to clarify for readers. To begin with, Apple's OS X is not directly derived from FreeBSD; it's derived from Darwin which was originally based on OpenStep 4.x and 4.4 BSD Lite (the common base for all BSD Unix projects). Later it incorporated some changes to the BSD code made for FreeBSD 3.2, specifically taking some command and library updates from the FreeBSD project. Today's Darwin takes some code from the modern FreeBSD project, but it still uses the Mach kernel, making it binary incompatible with FreeBSD. From all frames of reference, Darwin and FreeBSD are

two different operating systems with some common code between them. A new release of FreeBSD means absolutely nothing to OS X development because RELEASE is merely a tested snapshot of CURRENT, but the development of FreeBSD does further the development of Darwin and therefore OS X.

Secondly, the STABLE classification does not mean what it implies. STABLE is actually part of the development branch of FreeBSD. Although I mentioned this in the overview above, I'll restate for the sake of quotability: STABLE is where mature CURRENT code goes before it is accepted as part of the next production release.

Documentation

The FreeBSD Handbook is one of the best Free Unix documentation projects in the industry. It's well-organized, easy to read and understand, and generally up-to-date. Nearly every facet of installation and administration is covered in a simple and concise manner. There are also, of course, the standard Unix manual pages which are accessible from the command line. If you prefer a paper edition of the Handbook, you can buy one for $50 from the BSD Mall.

Installation

The installation procedure isn't any different from the way it was in 5.1 except that serial ATA drives are now recognized on most controllers. I personally tested it out on the VIA integrated controller on the Asus K8V Deluxe motherboard, but the hardware compatibility list has more information on other SATA controllers.

There are two discs to every modern FreeBSD distribution, or alternatively you can use a much smaller third disc (called miniinst) to do a minimum installation with just the base system (no packages). The first one is bootable and contains the base system and precompiled binary packages that were available when that version was released. Disc 2 is also bootable and has tools for assisting with system recovery in the event that your computer is unable to start due to kernel or configuration errors. Also available for download is a boot-only CD, which is exactly what it sounds like.

The FreeBSD installer (called sysinstall)is easy to use and navigate and it didn't have any trouble with any of the hardware I gave it. It's ncurses-based, so the menus are all designed in colored text and render perfectly on any video card and monitor. You can choose to use the CD's files to install your system or you can use the FreeBSD FTP site which will undoubtedly include updated packages and a newer Ports tree, but the FTP route will take longer to install.

The installer is fairly intuitive and informative, and everything works perfectly except for the built-in XFree86 configuration. I don't recommend testing your X Server through sysinstall as it is a sure way to crash the installer, forcing you to restart the entire installation process.

Installation time depends on whether you choose to install any of the packages and if so, how many you install. A typical installation geared toward desktop use will take less than an hour from start to finish, but an experienced FreeBSD user who has a more specialized and specific use for the system (such as for a firewall or server) can have everything up and running in fifteen or twenty minutes.

The FreeBSD bootloader, while simple and unable to be manually configured, is surprisingly useful. It automatically checks the IDE chain at boot time to see if there are any other bootable hard drives and gives you the option of starting from them instead of the FreeBSD disk. In other words it's dynamically configured, unlike GRUB or LILO which have to be manually adjusted in order to work correctly. This is a great advantage because it allows you to keep the boot records of other drives intact, enabling you to more easily create a multi-boot system. If you don't want to use the FreeBSD bootloader you do have the option of installing and using GRUB, or you can choose to not to have a bootloader at all.

Installing programs through the ports system is easy, as mentioned above. But for slower systems that need a lot of time to compile larger programs, it's easier to simply download a precompiled binary package from the package system. FreeBSD has many binary packages which are just as easy to install as they are through Debian's APT program. Precompiled packages and installed-from-source ports can easily work entwined in the same system. You can, for instance, compile XFree86 from source and then install KDE via the pkg_add command. Dependancies are, of course, automatically calculated and installed for you.

Updating

FreeBSD has quite possibly the best updating procedure in the industry. Keeping a FreeBSD system up to date is not simple or automatic (unless of course you write a script to do it for you), but it is easy and efficient and if it breaks it's able to be quickly fixed. The system is divided into two areas: the ports tree (which contains the source code and patch sets for all of the programs ported to FreeBSD) and the source tree (the core OS, including the userland and kernel). The two are updated separately to allow the user to keep their stable base system as it is while allowing installed programs to be updated to newer revisions. If your base system is working well and has all of the functionality you need, it's best to just leave it alone until you have a good reason to update it. Updating a working operating system can have negative consequences, as I've discovered once or twice.

The source and Ports trees are updated through the cvsup program, which connects to the FreeBSD CVS server and downloads updates and changes. Only the parts that have been changed are downloaded, and it shows you which specific ports or parts of the base system have been modified since your last update. This doesn't mean anything to your compiled system though, so if you want the changed source to be implemented you'll have to recompile the kernel and the userland and install them, a process that can take anywhere from one to several hours depending on system speed and other various factors.

Compiling just the kernel is quite easy: you edit the configuration file to your liking (click here to see what a typical kernel configuration file looks like) and then you're four commands away (three of which are often combined in one line) from a compiled and installed kernel.

As mentioned previously, the Ports tree is also updated through cvsup. Alternatively if you have other options set in your /etc/rc.conf configuration file you can run make update in the /usr/ports/ directory and it will do basically the same thing. Again this has no bearing on installed programs until you decide to install the updates by recompiling the updated programs. Usually after you update the ports tree you'll want to update the database that keeps track of your installed programs; this is done with one simple command, and it checks for stale or circular dependancies, changed

package names, and stale origins. Next, the portupgrade command takes care of all of the downloading and compiling of updated programs for you. The reason why this is a three-step process is to make the updating procedure more reliable and easier to fix. Having survived the nightmare of Gentoo Linux's always fatally broken and never easily fixed Portage system, I can tell you that "ease of use" means "difficult to fix" because it doesn't allow the user to control the process. I've seen APT and Portage choke on dependancies with no obvious way to fix them, and anyone who has ever tried to use a third-party RPM knows what a disaster that can be. FreeBSD is, if nothing else, a nice respite from the various GNU/Linux package management systems.

There is a binary update utility for FreeBSD currently in development, and a separate utility to perform binary security updates, but it does not yet work with 5.2-RELEASE.

Features

There are few meaningful differences between 5.2 and 5.1, most of them being small changes to various userland commands. A complete listing can be found here in the release notes. The most drastic changes in 5.2 are:
Client support for NFS version 4
Full tier-1 support for single and multi-CPU AMD64 systems
Improved driver support for IDE, SATA, and 802.11a/b/g devices, and significantly better integration with the ACPI power management subsystem
Dynamically linked root partition
Experimental first-stage support for multithreaded filtering and forwarding of IP traffic.

Client support for NFS version 4 means that FreeBSD now can access NFSv4 shares, which can have stronger security and support traditional file access with file locking and the mount protocol. NFSv4 also supports internationalization, client caching, and compound operations, all of which were not available in previous NFS versions.

Full Tier 1 support for AMD64 means that it is production quality and fully supported by the security officer. This is the first release of FreeBSD to have full Tier 1 support for AMD64. It actually works quite well with AMD64 hardware, but there are some problems: Linux binary compatibility and IA32 binary compatibility don't work yet. That means that the only programs you can install and run are the ones that specifically support the AMD64 architecture. As of 1/21/04 that means no CVSup (although there is a hacked CVSup available which works reasonably well) and no Java support, which also means that programs which require Java will not compile or run. While most of the programs in the Ports tree will work, the ones that don't are pretty important. I've also noticed that mouse support is somewhat limited for some reason -- my Wireless Microsoft Intellimouse Explorer 2.0 and Logitech Cordless MX 700 mice don't work or work very poorly in the AMD64 edition, but in the i386 edition they work just fine.

As previously mentioned, there is now support for SATA hard drives and a wider array of wireless LAN cards. Better ACPI support has been added; the kernel now complies with ACPI 2.0 standards.

The root partition is now dynamically linked, which improves integration with the NSS (Name Service Switch) subsystem and reduces the installed footprint of the base system.

The initial steps have been taken to make the entire network stack fully multithreaded, which would significantly improve its efficiency and performance in SMP systems.

FreeBSD also features a large development team that values good code over expanded features, a cohesive base system that is developed as one unit instead of a separate kernel and userland, a friendly and helpful community, excellent documentation, and over 10,000 ported software applications.

Bugs

Most of my testing was done on the AMD64 edition, but I did install and use the i386 edition as well.

The first thing I noticed when I upgraded to 5.2 was that dhclient, the utility that starts DHCP services, no longer worked correctly. My motherboard has integrated 3Com gigabit LAN using the SysKonnect chip; in 5.1-CURRENT (as of the middle of November) everything worked perfectly... then sometime in December someone changed the CURRENT code somewhere and boot-time DHCP services started less than half the time. With 5.2-RELEASE it doesn't work at all. Changes were made to both the sk driver and dhclient, but I can't figure out which one is at fault. I figured out a long-winded fix for the problem, which is to quit dhclient by using the -r switch, then start tcpdump to send the card into promiscuous mode, then ctrl-c out of it and start the /etc/netstart script to start DHCP. I have to do this every time I boot the machine. I tried some patches that were available but none of them worked; in the end I filed a bug report to ensure that the problem was addressed.

As previously mentioned, the AMD64 edition has trouble with at least two mice. The XFree86 packages don't come with the same video card support that the same programs in the Ports system do. Specifically the precompiled packages didn't have VESA driver support, which is an issue for me because it's the only driver that works for the ATI Radeon 9800 Pro right now.

I ran into a mysterious bug with KUser which deleted my root password... the only solution to this problem was to reinstall the base system from the CD. The problem was addressed in KDE 3.14, and I had installed the KDE package from FTP which was apparently out of date. In fact, package installation was nothing short of disastrous for me. The alternative is of course the Ports system, which I prefer, but it took a while to get the right /etc/make.conf options. Namely, I needed to compile a lot of programs with -fPIC.

It seemed to me that there were far more problems with 5.2-RELEASE than there with 5.1-RELEASE, and actually I was doing much better with my 5.1-CURRENT build from November.

Conclusions

Now that I've patched the right files and found the right compile options and workarounds, I can use FreeBSD again. It's not a pleasant experience to have to use other operating systems once you're used to FreeBSD, and there was a period of about a week that I was unable to use it at all because I couldn't get back to the 5.1-CURRENT build that worked so well for me and I hadn't resolved the DHCP problem. Overall I would say that 5.2-RELEASE should have gone through at least a few more rounds of testing and bugfixing before it was released.

Most disappointingly there was no added support for SoundBlaster Audigy and Audigy2 sound cards even though a working patch (several, actually) has been available for over a year. According to CVS a new and improved Audigy driver was finally committed shortly after 5.2 was released. So here I am still patching just to get my sound card to work, solidifying the suspicion that 5.2 was released too early. I tried to download the changed sound driver files from CVS and compile my kernel with them, but there were errors -- and upgrading to CURRENT caused the system to hang at startup because of the infamous ATA driver problem which was absent from 5.2-RELEASE but has come back for revenge in CURRENT.

Overall 5.2-RELEASE is disappointing from a desktop perspective, but it's still more advanced than any community GNU/Linux distribution that you'll find, especially in the area of AMD64 support. From a networking perspective there have been a lot of improvements (minus the sk/dhclient bug) that can add a lot of functionality and enhance performance. This is, after all, the "new technology release" and as such it is not meant for production servers, but I was expecting more from 5.2. I'm left wondering what the point of this release was, seeing as how there are new bugs and few significant improvements to the base system.

At this point I feel a 5.2.1-RELEASE is in order... if for nothing else, at least to fix the DHCP problems with the sk driver (or is it a problem with dhclient?).

If you're using 5.1 right now and you're happy with it, my advice is not to upgrade to 5.2 unless you're willing to wrestle with it a little and possibly be forced to install the whole operating system again. If you're looking for a free (as in money) operating system with good AMD64 support, FreeBSD is the best you'll find right now. It's missing Java support, Linux and IA32 binary compatibility, but at least two of those three things should be added to CURRENT sometime in the near future. Most of the programs in the Ports system now work well with AMD64.

This article is re-printed with permission. The originals can be found at:

http://www.thejemreport.com/modules.php?op=modload&name=News&file=article&sid=108&mode=thread&order=0&thold=0

Book reviews

Author: Grant Allen

Perl for Oracle DBAs
Authors: Andy Duncan and Jared Still
Publisher: O'Reilly & Associates
ISBN: 0-596-00210-6

The greatest problem when automating the world of Oracle database management and monitoring is finding the time to build the tools. If you're like me, you spend your few idle minutes of the day thinking "Next time I have a spare moment, I'll write a little utility to make this job easier". Well you can stop day dreaming – the reality has arrived.

A wealth of ready-for-use tools are served up for your pleasure in O'Reilly's most recent Oracle tome – Perl for Oracle DBAs. The authors, Andy Duncan and Jared Still, deliver a smorgassboard of tools and utilities developed with and around Perl, providing relief for the distraught DBA – and a good dose of humour to boot!

As is normal with O'Reilly books, the topic is split into digestible parts; in this instance, three in logical sequence. While each part could be read in isolation, the easy flow of the sections will aid Perl novices.

Part one gives the now-ubiquitous but enjoyable biography of Perl, its development by Larry Wall, and the growth and dynamism it has embodied. After a brief synopsis of Perl's merits, a discussion of the main Perl-to-database connectivity layer – the Perl DBI – follows, illustrating how Perl connects to, and works with, Oracle and other databases. The pace quickens, with a thorough step-by-step approach to sourcing and installing Perl as the foundation for the later parts.

Andy and Jared deserve credit for handling this task well. No assumptions are made regarding "preferred" environments, and no shortcuts are taken. Instead, detailed advice is given for obtaining and deploying Perl source and binaries under Unix, Linux, Windows and Cygwin environments. This includes an introduction to the Comprehensive Perl Archive Network – CPAN, and the ppm facility under Windows. Regardless of your chosen platform, you'll find the clear, concise approach to the "download; unzip; make; install" discipline takes any ambiguity out of the process.

It's hard to do justice to the second part of the book in a short review. It provides an introduction to no less than eight fully function Perl suites providing various Oracle management features. These include well-known apps like Senora and Orac, to tools that have wider application that just Oracle database management, like DBD::Chart.

The book then takes a slight tangent that at first sight looks distracting, discussing web extensions for Perl; in particular Apache's mod_perl as the in-process Perl interpreter, and it's advantages over CGI. Had I not been so eager to devour the book's content, I would have noted that this was a necessary entree to the joys of Oracle's PL/SQL web toolkit, and the advantages of embedding Perl into web pages for Oracle management using Embperl and Mason – covered in the latter half of Part two. Before moving on to the final part, we are exposed to the relatively new Perl module, Oracle::OCI, which provides a low-level one-to-one wrapping of Oracle's native Oracle Call Interface in Perl.

My only criticism of the book is that I would have liked more detail on some of these tools. That may be a little harsh, as I'm sure the sub-editors were keeping a tight rein on a book that already runs to 600 pages.

Part three is the perfect culmination of the earlier parts' build-up. We are introduced to the Perl DBA toolkit – PDBA – which in the author's own words is

"... a set of Perl scripts and reusable modules that we're developed to help Oracle DBAs perform both routine database administration tasks and more advanced monitoring and tuning."

As active members of the Oracle DBA community (and in Jared's case, moderator of the influential Oracle-L mailing list), the authors' are at the leading edge of contemporary management ideas for Oracle databases, and they have selflessly donated what amounts to an excellent management framework to their peers in releasing PDBA.

Ever worried at night about the fact cron or at jobs are running important batch and housekeeping work, but with passwords accessible to the wily hacker? Then the PDBA password server is for you. If you've had the the joy of dealing with users questioning why you can't just add 3000 user accounts by the end of the day, you'll love the create_user modules. Plenty of other pet DBA hates are covered: Tablespace free space and extent monitoring; Dead connection detection and clean-up; even a monitor to watch the other monitors! The book is rounded out with a handy collection of appendices covering Perl, the Perl DBI, regular expressions and Perl data munging.

In short, O'Reilly's Perl for Oracle DBAs is a great asset for any DBA looking to get on with the job – both Perl novices and old hands.

Book reviews

Author: Tony Davies

TOMCAT: THE DEFINITIVE GUIDE.

While working as a systems administrator at a local ISP I had the unfortunate experience of having to setup a Tomcat server. I can not count the hours I spent trying to figure out how to make the thing work. Jakarta's documentation on configuring Tomcat from the ground up is sorely lacking and extremely frustrating. I ended up getting the clients website up and running but the boss was not pleased, not to mention the headaches of adding a 2nd client's website later on. I wish I had this book back then.

"Tomcat: The Definitive Guide" fills the void that Tomcat's documentation creates, explaining the configuration and setup of Tomcat nicely, covering such topics as building Tomcat from sources, integrating Tomcat into Apache using mod_jk2, setting up tomcat to authenticate users and sessions using realms, roles and users, deploying web apps, and performance tuning.

Integrating Tomcat with Apache was probably the most useful chapter for me personally. I have moved from being an admin to a developer, I no longer want to spend time reading configuration documentation so when I get a new client I can integrate their setup into ours. I am now selfish; I only care about getting the server setup for me, so I can start cutting code, someone else can worry about integrating my setup into their server.

The chapter on deploying web applications I also found interesting. I guess I am not as selfish as I first thought, after being there, I can appreciate the frustration of people wanting custom configuration in a global configuration because they were unaware of how to do it inside their own web app. This chapter helped me learn how to do this for my self, and then package it up in a nice little war file so that the administrator at the other end can just dump it in a directory and restart Tomcat.

Having a chapter on the 4 main configuration files was also a big help. A description on each of the elements available was extremely useful and thankfully more verbose then the documentation that comes with Tomcat and will make a useful future reference.

Large scale Tomcat Administrators will find the chapters on performance tuning and server clustering helpful but since I am developing at most 3 applications at a time, this wasn't really of benefit to me.

Working for an information an IT Security company the chapter on securing Tomcat was extremely valuable. This chapter discussed topics ranging from running Tomcat in a chroot jail to securing information transmitted and received using ssl.

"Tomcat: The Definitive Guide" is definitely geared more towards a Systems Administrator, so that they can get the most out of their Tomcat server and would make a worthwhile addition to their bookshelf. As a JSP/servlet developer who still dabbles in a bit of admining, I also found parts of this book worth reading.

Comments on OSS/FS Software Configuration Management Systems

Author David A. Wheeler:<dwheeler@dwheeler.com>

With the release of Subversion 1.0, lots of people are discussing the pros and cons of various software configuration management (SCM) / version control systems available as open source software / Free Software (OSS/FS). Indeed, the problem is now an embarassment of reasonable choices: there are several OSS/FS SCM systems available today. Here's some information about SCM systems that I've learned that you may find helpful; I'll discuss three popular choices (CVS, Subversion, and GNU arch), the differences between centralized and decentralized SCM, using arch to support centralized development, and a few links to other reviews. Feel free to also look at my paper on SCM security.

CVS, SUBVERSION, AND GNU ARCH

In my opinion three OSS/FS SCM systems get the most discussion: CVS, Subversion, and GNU Arch. There are certainly others, and I don't mean to intentionally exclude them, but I just haven't had the time to examine the others in as much depth (Monotone, in particular, looks very interesting). Besides, knowing about these three will help you understand the rest. So, here's a brief discussion about each:

CVS is extremely popular, and it does the job. However, it's showing its age through a number of awkward limitations: changes are tracked per-file instead of per-change, commits aren't atomic, renaming files and directories is awkward, and its branching limitations mean that you'd better faithfully tag things or there'll be trouble later. The CVS maintainers have also declared that the code has become too crusty to effectively maintain. These problems led the main CVS developers to start over and create Subversion.

Subversion (SVN) is a new system, intending to be a simple replacement of CVS. Subversion is basically a re-implementation of CVS with its warts fixed, and it still works the same basic way (supporting a centralized repository). Like CVS, subversion by itself

is intended to support a centralized repository for developers and doesn't handle decentralized development well; the svk project extends subversion to support decentralized development.

From a technology point-of-view you can definitely argue with some of subversion's decisions. For example, they don't handle changesets as directly as you'd expect given their centrality to the problem. But technical advancement is not the same as utility; for many people who currently use CVS and just want an incremental improvement, subversion is probably more or less what they were expecting and looking for. But there are weaknesses, for example, Subversion doesn't keep track of "which patches have already been applied" on a given branch, and trying to reapply a patch more than once causes problems. Thus, subversion has trouble with history-sensitive merging of branches where the branches share parts (GNU arch doesn't have this problem, because it does track what merges have been applied). There have been concerns about Subversion's use of db to store data (rather than the safer flat files), since in a few cases this can let things get "stuck". In practice this doesn't seem to be so bad (in part because the data can be extracted), but certainly some are concerned.

Subversion uses a BSD-old-like license that, while OSS/FS, is GPL-incompatible, and that's unfortunate (GPL incompatibility can be a problem). Subversion can be used to maintain GPL software or any other kind, without restrictions. Subversion depends on a large number of libraries and programs (and can be perceived as rather "heavyweight"), so it can take some effort to install currently; distributions will probably be quick to include it, so that problem should go away relatively soon. This book on Subversion gives more information about it.

If you're using CVS and want a simple upgrade path to something better, Subversion appears to be the simplest approach. It works in a very similar way to CVS (in particular through a centralized repository), allowing any of the authorized developers to immediately modify a shared repository (with a record that it was done so and rollback capability). Subversion is what it intends to be: an improved CVS.

GNU arch is a very interesting competitor, and works in a completely different way from CVS and Subversion. GNU arch is fully decentralized, which makes it very work well for decentralized development (like the Linux kernel's development process). It has a very clever and remarkably simple approach to handling data, so it works very easily with many other tools. The "smarts" are in the client tools, not the server, so a simple secure ftp site or shared directory can serve as the repository, an intriguing capability for such a powerful SCM system. It has simple dependencies, so it's easy to set up too.

Decentralized development has its strengths, particularly in allowing different people to try different approaches (e.g., independent branches and forks) independently and then bringing them together later. This ability to scale and support "survival of the fittest" is what makes decentralized development so important for Linux kernel maintenance. Arch can also be used for centralized development, but see my discussion below about that.

Indeed, I really like arch, yet I'm also frustrated by it. It has so many positive strengths, so it might be confusing why I think it has some problems. So, here's a discussion of its problems, which basically show GNU arch is a tool that's already very usable but needs some maturing.

A serious weakness of arch is that it doesn't work well on Windows-based systems, and it's not clear if that will ever change. There are ports of arch, both non-native (Cygwin and Services for Unix) and a native port too. However, the current win32 port is only in its early stages, and the Win32 page on the Arch wiki says "Arch was never intended to run on a non-POSIX system. Don't expect to have a full blown arch on your Microsoft computer." At least part of the problem is the long filenames used internally by arch; arch could certainly be modified to help, though there doesn't seem to be much movement in that direction. Other problematic areas include symbolic links, proper file permissions, and newline problems, as well as the general immaturity of the port as of March 2004. Some people don't think that poor Windows support is a problem; to me (and others!), that's a serious problem. Even if you don't use any Microsoft Windows systems, people don't want to use many different SCM systems, so if one can handle many environments and the other can't, people will use the one that can handle more environments. I think GNU Arch's use will be hampered by this lack of support as long as this is true, even for people who never use Windows; good native Windows support is very important for an SCM tool.

As of February 2004 Arch has some awkward weaknesses involving filenames. It still can't handle spaces in filenames, a significant defect (though this is finally scheduled to be fixed soon). More fundamentally, it uses extremely odd filenaming conventions that cause trouble for scripts, command-line use, and many common tools. Its "+" prefixes cause problems with extremely common tools like vi, vim, and the pager more (this is especially a problem when trying to enter change log information - why choose a convention that's inconvenient for one of the world's most popular text editors?). Its "=" prefixes expose a bug in bash filename completion (this bug will eventually be fixed in bash, but buggy implementations will be around for a long time to come because this is such a rare need and bash is the default shell for many systems). And although this is less of a problem, it stores data in an "{arch}" directory, but the "{}" characters cause problems for many shells (particularly C shells) because they have a special meaning (they're filename globbing characters like "*"). For example, in C shells you can't "cd {arch}" or "vi {arch}/whatever"; you must quote the directory name. The problem isn't that filename conventions are a bad idea; most CM systems have them! The problem is that some of the conventions chosen by arch seem to be designed to interfere with commonly-used tools, and thus require using many work-arounds when using common tools (such as prefixing the filename with "./" or using the "--" option). That's unfortunate since GNU Arch's underlying concepts work well with other tools; if the developers had chosen better conventions these problems would never have occurred. I suspect these poorly-chosen conventions are too ingrained to be easily changed now, but there's always hope. There are ways to override the defaults in some cases, but not in many, and tools should choose good defaults. It's too bad, because nothing in arch's fundamental design requires these particular filename conventions.

GNU arch gives you a lot of control using lower-level commands, but it doesn't (yet) automate a number of tasks that it really should be automating. Many common operations require multiple commands, when instead a single command and reasonable options should be enough for most people. If you use a single archive for a long time in GNU arch, it eventually accumulates a very large amount of data and becomes inconvenient to work with. arch's developer suggests dividing archives by time and including a date in the archive name. I think handling this accumulation is a nuisance; this kind of manual work is exactly what an SCM should handle

automatically (e.g., perhaps arch could hide branches that have been unused in more than a year, by default). Arch has nice caching facilities, which can speed access to specific versions, but a cache has to be created by hand (by default the tool should automatically create caches, and remove old automatically-created caches, as well). Arch works slowly if the {arch} directory is on NFS; the tool should be able to detect slow execution and automatically try to find an efficient alternative, instead of requiring user workarounds. Many arch developers seem to create a similar set of higher-level specialized scripts to automate common tasks, but that's missing the point: you shouldn't have to write scripts to make a tool automate common tasks. An SCM tool should include commands that, through automation and good defaults, "do the right thing" for common tasks. The good news is that the arch developers are realizing that this is a problem and correcting it. The "rm" (delete) command deletes both the id and the corresponding file automatically (instead of requiring two steps); that capability was only added on February 23, 2004, though, so clearly automating steps has only begun. The documentation notes that automatic cache management is desirable; it just hasn't been done. The mirroring capability is clever, but if you download a mirror and make a change, you can't commit the change and the tool isn't smart enough to automatically help (even though the tool does have information on the mirror's source). The website described a complicated workaround using undo and redo, and Jan Huldec described a simpler approach (using tag, sync-tree, and set-tree-version), but the tool should be able to help commit changes even if you downloaded from a mirror.

Arch will sometimes allow dangerous or problematic operations that just shouldn't be allowed. For example, branches should be either commit-based branches (all revisions after base-0 are created by commit) or tag-based branches (all revisions are created by tag); merging commands will not work otherwise, yet the tool doesn't enforce this limitation. The tla tool doesn't check if there are still pending merge rejections (.rej reject files), so operations such as commit, update, replay, or star-merge produce a scrambled workarea; users make mistakes, and an SCM system should work to protect data.

The user interface also has some problems. Under the user nightmare clause, the "mv" and "move" commands do different things: "mv" moves moves both the id and the file, while "move" only moves the id. This user interface seems designed for confusion; why not make "move" and "mv" the same, and make "mv-id" the only command that only manipulates id's? Many commands are aliases, which simply makes documentation unnecessarily complicated.

The arch documentation is weak and needs more work; that's especially unfortunate, because the documentation issues can hamper early adopters who want to start using it today. A careful reading of what's available on-line should be enough for at least basic use of arch, though. Much of the documentation emphasizes lower-level implementation details (e.g., exactly how a command is implemented in the local filesystem) instead of emphasizing the higher-level constructs. Some of the documentation emphasize aliases, which is extremely distracting; if "add" and "add-id" mean the same thing, just document "add" (and later on, in an ignorable note, list the aliases). In some cases the documentation needs to be updated for what the software actually does. The on-line tutorial at the FSF GNU arch website is a good place to start, and the Arch Wiki is an especially good place to find some more detailed reference material.

In general, GNU arch isn't currently as mature as subversion. Its implementation needs more shaking down, its weird filename limitations should be fixed, and it sometimes requires users to do optimizations "by hand" when the tool should be handling it automatically. As noted above, its commands are sometimes on the low-level side; it can take several simple commands to set up values that should be defaults or built-in recipes/commands. And the documentation needs work.

But don't count out GNU arch for the long term based on these problems, most of which are short-term. Many of these problems simply reflect the fact that GNU arch hasn't had as much time to mature as other tools like subversion. I'm documenting these problems because, in fact, GNU arch has a lot going for it. In my opinion, the GNU arch developers have emphasized simplicity, openness of design, and power (ability to handle complex situations), and have paid less attention so far to ease of use (especially for simple situations). Thus, although it has problems as noted above, GNU arch is extremely powerful and its basic concepts are very flexible. More time and tools that build on top of GNU arch can resolve these issues. Arch is also endorsed by the Free Software Foundation (FSF) and directly supported by their Savannah system; that's certainly no guarantee of success, but endorsements like that often bring users and developers to a project, increasing its likelihood of success. GNU arch is a frankly more interesting approach to the problem, and it has a lot of promise.

Centralized vs. Decentralized SCM

As you can tell, there seems to be two different schools of thought on how SCM systems should work. Some people believe SCM systems should primarily aid in controlling a centralized repository, and so they design their tool to support a centralized repository (such as CVS and Subversion). Others believe SCM systems should primarily aid in allowing independent developers to work asynchronously, and then synchronize and pull in changes from each others, so they develop tools to support a decentralized approach (like GNU arch, monotone, darcs, and Bitkeeper). Tools built to support one approach can be used to support the other approach, but it's still important to understand the difference.

Tools built to support one camp can sometimes support the other approach, to at least some extent. Conceptually a distributed approach should be able to fully implement the centralized approach without too much trouble. However, it's not as clear to me that these supports for the "other approach" are always as good as a tool made to do the same thing natively, particularly when centralized systems try to support decentralized development. Subversion has svk, which builds a distributed SCM system on top of subversion. However, implementing svk on top of subversion is a very heavyweight way to create a distributed SCM system, far exceeding what it takes to implement a natively distributed SCM system. GNU arch can easily support a centralized repository by having developers share read/write privileges to a directory that implements the repository, but see the discussion below about security concerns I have (due to the direct control over the repository by users). There's also the extra tool arch-pqm which can help mitigate some of my security concerns, though it's not currently integrated into GNU arch. The various projects' supporters all seem to feel that "their side" does adequately support the other approach, though. I do expect that the different projects will continue working to get better at supporting the "other" approach, so in a few years this distinction may get really fuzzy.

A posting by Bastiaan Veelo at Linux Weekly News has a nice

summary:

"The most important thing to be aware of though is that Arch and Subversion differ in fundamental ways. Arch works in a decentralized way, while Subversion is designed on a client/server model. Indeed with Arch you can start coding and using version control without first applying for access to the server. However, [merging] your code with the main branch has to be done by the one project maintainer....

Development with Subversion (and CVS for that matter) is centralized in the sense that there is just one repository, but it is actually more decentralized in a social sense since there are as many code integrators as there are developers with write access to the repository.

In short, one could say that Arch is centralized around a code integrator, and that Subversion (like CVS) is centralized around a repository. You decide what fits best. If you are a heavy user of CVS... chances are that Subversion actually fits your needs best.

Using Arch to Support Centralized Development

As I noted above, conceptually a distributed approach should be able to fully implement the centralized approach. I do have some concerns about the recommended method for using GNU arch to support a centralized repository of multiple developers. It appears that some support tools will deal with my concerns, though using them takes much more effort.

The GNU Arch wiki site provides basic information on how to use arch in a centralized way. It's easy to use GNU arch to implement a centralized repository: a particularly simple way is to grant all developers read/write access to a shared filesystem (say secure ftp) used to create the centralized repository. The "repository" is in some sense a pseudo-user that everyone can write to. Systems hosting many project repositories that need to be protected from each other will need to define users or groups (say one per project) to provide that separation. This can viewed as a minor problem (now the system administrator or a special group management tool needs to get involved whenever a new project or new developer joins a project) or a big plus (operating system controls are heavily tested and far more reliable than application-level access controls). Once set up, there are certainly many advantages to this scheme. For example, it's often easier to set up a shared directory than a more complex server.

However, I think there are problems when using arch this way. This approach presumes that all the clients "work perfectly;" if there are many developers, the odds increase that some developer is using an older client with a bug or subtle semantic difference that could screw up the whole repository. More importantly, it presumes that developers, and attackers who temporarily gain developer privileges, are never malicious. Since a developer has complete unfettered read/write access to a shared repository, a malicious developer (or attacker taking the developer's credentials) could stomp over a shared arch repository, changing supposedly unchanging data to make the repository quite different than expected. Unless there's something to counteract it, a malicious developer or attacker with their privileges could insert malicious code without making it clear that they inserted it, make it appear that some other developer inserted malicious code, or erase data in a way that makes it unrecoverable. Obviously, malicious developers are bad thing, but an SCM system should always be able identify exactly who inserted any malicious code (in a nonrepudiable way), and protect the integrity of the SCM history so that changes can be easily undone (and re-checked, once you've found a culprit). In today's unfriendly world, where you're often working with people you don't really know, protection against malicious attack is important.

The recommended GNU arch setup for a central repository has all users sharing a single account, so the operating system and arch have no way to even distinguish between the users when they log in! It's possible to set up a shared directory repository so that users authenticate individually, and then set up a shared directory (using groups), but users can then accidentally (or intentionally) set their access control bits so that later developers won't be able to read or modify the files. So, the recommended approach has a lot of drawbacks if a client misbehaves, or you don't fully trust your developers, or an attacker might gain developer privileges.

You can make backups and compare them with the original, which would at least detect malicious changes to the repository history if they happen after the backup. Backups would also allow people to replace the malicious change with the correct version. Note, however, that arch doesn't currently include tools to do this checking automatically (I don't think you can use arch's mirroring capability, since the arch data itself is suspect). So, you'll have to know a lot about arch's internals to do this currently, until arch adds such tools. This approach would not identify exactly who made the malicious change, even when the culprit could have been required to log in as a specific developer. But possibly more importantly, a malicious developer could trivially create a malicious change and forge it as though someone else made the change. A backup could only tell you that an addition had been made, but it can't say if the data in the addition is correct. So backups definitely help, but attackers can get around them.

Another partial (but significant) counter to these problems are the new signing archives capabilities added to arch 1.2. You can optionally make an archive a "signed" archive, in which the changes are cryptographically signed. I've looked into this (my thanks to Colin Walters who helped me understand details of the signature process). When enabled arch can sign MD5 hashes, which are cryptographically much weaker than SHA-1 hashes, but that's certainly a step forward from having no cryptographic signatures. Some effort is definitely required to set up signed archives (e.g., now you need public keys of all developers), though it's a good idea for security-minded systems. The signatures sign the revision number as well as the change itself (they're both encoded in the signed tarball), so an attacker can't just change the patch order and can't silently remove a patch and renumber the later patches without detection. However, it appears to me that such signatures (at least as currently implemented) cannot detect the malicious substitution of whole signed patches (such as the silent replacement of a previous security fix with a non-fix), or removal of the "latest" fix before anyone else uses it. Unlike backups, signatures can detect many problems without comparing an external source (so it'll likely be faster to detect problems), and it's built-in to the tool already, which increases the likelihood it'll be used. For many developers, backups and signing archives may be enough. However, this mechanism still doesn't expose who made certain kinds of malicious changes (such as silent removal and replacement), in the case where the developer could have been identified.

Arch-pqm (patch queue manager) is an arch extension that creates a central repository out of a decentralized tool. It allows developers to send their requests (such as changes) to a central location, then arch-pqm queues up those requests and has them automatically

performed. Arch-pqm first checks the GNUPG signatures of the requests to determine if the requester is an authorized developer for that repository, and rejects changes by anyone else. This is closer in approach to how centralized tools like CVS and subversion work. I've had several email conversations with arch-pqm's developer, Colin Walters, and found that arch-pqm only permits operations that protect the history of the repository. In particular, arch-pqm supports the star-merge operation to merge in new changes, caching, uncaching, making new categories / branches / versions, and tagging -- none of which erase the history in the repository.

Thus, it currently appears to me that combining signed archives, backups, and arch-pqm will probably address my concerns. Arch-pqm prevents arbitrary developers, who have rights to the repository, from arbitrarily changing the frozen repository values. Signed archives and comparisons with backups allow the detection and repair of malicious changes to the repository if the attackers work around or subvert arch-pqm. If a malicious developer's changes can always be recorded correctly as theirs and undone later (by forcing them to sign their changes), and at least detected when the infrastructure can't do otherwise, then my concerns disappear. One caveat: I haven't done a detailed security analysis, and arch-pqm wasn't originally designed specifically to provide this security. For example, perhaps creating odd filenames or trying to change settings might subvert this protection. There may be ways to create to exploit a buffer overflow or other technique to subvert these checks. Still, the basic concepts seem sound, and some security analysis at least has a chance with this setup. Unfortunately, using arch-pqm isn't yet built into arch, and the backup checking isn't built into arch either, so there's more than a little "rolling your own" effort to implement and use this approach. Also, the documentation doesn't lay out a simple step-by-step method for setting it up.

I should note that currently I don't think Arch supports signing of signatures. In other words, if B accepts A's work, and C accepts B's work (which included A's work), then I should see signatures by A of A's work, and signatures of B indicating that they accepted A's work. To be fair, few SCM systems support that. But centralized systems have an easier time providing equivalent functionality; distributed systems should record more of this kind of information, because there's no central place to get it or trust it.

Note that Colin Walters is also creating a "smart server" for arch named "archd" and a protocol to support the server. In some ways this appears to be similar in concept to arch-pqm; it would be a program that would automatically execute SCM commands from authorized users. However, archd would use a specialized protocol designed for the purpose to transfer the data, rather than using email. It appears that it will have similar protections (it will limit the commands that can be executed), and if that's true, the same comments would probably apply. But this would be for the future; it's not ready for use at this time.

In all SCMs, if you're worried about malicious developers, you have to be careful about who can define "hooks" and the permissions they have when they run. Whenever GNU arch runs a command, GNU arch runs the program ~/.arch-params/hook (if it exists) to run additional actions ("hooks"). In other words, the hooks are defined on a per-user basis, not per-project basis. That design has some advantages from a security point-of-view; since the hook is not inside the maintained development area (normally), editing files shouldn't trick the CM system into running new commands. However, that has disadvantages if there's a shared repository, because that means that the shared repository can't run commands to enforce some requirements (e.g., to require that there be no compiler warnings, run regression tests, announce a change via email, or require two-person authorization before checking in). This can also be solved by arch-pqm or a smart server, since the server can run the hooks on its own in its own environment.

Other OSS/FS SCM systems and Other Reviews

There are other OSS/FS SCM systems, such as Monotone, Aegis, Darcs, and Vesta. I'm not trying to completely exclude them from consideration; I just don't have enough time to analyze them too. You should certain investigate the various alternatives before picking an SCM system.

Monotone looks especially interesting, as it's different approach to a distributed SCM. As Shlomi Fish describes it, "changesets are posted to a depot (that can be a CGI script, an NNTP newsgroup or a mailing list), which collects changesets from various sources. Afterwards, each developer commits the desirable changesets into his own private repository.... Monotone identifies the versions of files and directories using their SHA1 checksum. Thus, it can identify when a file was copied or moved, if the signature is identical and merge the two copies. It also has a command set that tries to emulate CVS as much as possible." Monotone has recently fixed some of its problems in handling unusual filenames (this seems to be a common problem in SCM systems). Monotone's emphasis on security, and its clear concepts, make it another SCM worth considering. Monotone's approach is based on three-way merging and SHA-1 hashes. The Monotone folks argue that the Arch approach is somewhat weaker than Monotone's approach, but note that Monotone isn't nearly as good as Arch in supporting some kinds of "cherry-picking" (see the Monotone FAQ for more information), so it's hard for me to declare either one a "winner" in terms of merge capabilities. Monotone does appear to be less popular than GNU Arch (as determined by Google link counts), for what that's worth.

However, I didn't examine some SCM programs seriously because the little I learned suggested I should look elsewhere first. The better SCM initiative's information about Aegis convinced me that I shouldn't look hard at it. An Aegis user has since told me that Aegis is better than that review claims, so this may have been too harsh. The better SCM initiative claimed that Aegis requires running as root, which in my mind is an inexcusable security weakness that immediately turned me off. It also reported that it was very hard it is to install, which again made me not very interested in examining it further. I hope to take a further look at Aegis in the future. The same review reported that "Vesta is reported to be mature" but because only Vesta can be used to build Vesta, I expect that it'll be hard for it to attract new users and developers. RCS is much older (as is SCCS before it); its lock-based approach just doesn't work well with today's fast development cycles and large development groups. Bitkeeper is powerful, but it isn't OSS/FS, so it's outside the scope of this paper.

I will add a few comments about darcs. From what I've seen, darcs is currently more of a prototype of some very innovative ideas for SCM, and maybe a tool for smaller projects, rather than a useful tool for large projects, though could be used. Darcs is written in Haskell, which is a strength and a weakness. Haskell is a high-level functional programming language, which probably helped the developer concentrate on abstract concepts. However, while Haskell is intriguing, in my experience programs written in it are generally slow. Some have argued to me that Haskell isn't necessarily true today, and maybe that's true, but darcs' developer admits that darcs has poor performance (which would cause trouble as a project gets large), though in March 2004 he says it has gotten better. Since few

developers truly grok functional programming, darcs is less likely to get other developers to help extend it (it does get contributions, but nothing compared to Subversion or GNU Arc). Darcs' website stated that it does not have an "abundance of features" and its "core may be still be buggy" -- not exactly the words you want to hear when you let a program control your source code! The main developer does say that the website is out of date, that the program is no longer buggy, and that it supports more than basics (though it is missing some features). It does have some innovative approaches, though, and some of its concepts may slip into the next SCM systems. For example, darcs can keep track of inter-patch dependencies so that bringing in just one patch can bring in "just the others needed", a clever capability not supported by other tools like GNU Arch. It is completely patch-oriented, and requires user input to help characterize exactly what changed. For example, it understands a "token replace patch", which makes it possible to create a patch which changes every instance of the variable ``stupidly_named_var" with ``better_var_name", while leaving ``other_stupidly_named_var" untouched. As the author says, "When this patch is merged with any other patch involving the ``stupidly_named_var", that instance will also be modified to ``better_var_name". This is in contrast to a more conventional merging method which would not only fail to change new instances of the variable, but would also involve conflicts when merging with any patch that modifies lines containing the variable. By more using additional information about the programmer's intent, darcs is thus able to make the process of changing a variable name the trivial task that it really is..." The advantage is that merge conflicts can suddenly disappear; the disadvantage is that this requires more interaction with the developer, who already has a complicated problem. Whether or not this approach will catch on is to be seen; I doubt it, myself, since systems which don't have it seem to be acceptable to most developers.

There are many other SCM comparisons available. The better SCM initiative was established to encourage improved OSS/FS SCM systems, by discussing and comparing them. Among other things, see their comparison file. Shlomi Fish's OnLamp.com article compares various CM systems as does his Evolution of a Revision Control User. The arch folks have developed a comparison of arch with Subversion and CVS (obviously, they like arch). Another pro-arch discussion is Why the Future is Distributed. A pro-subversion discussion is available at Dispelling Subversion FUD. Slashdot had a discussion when Subversion 1.0 was announced. Kernel traffic posted a summary of a technical discussion about BitKeeper. Brad Appleton has collected lots of interesting SCM links. Zooko has written a short review of OSS/FS SCM tools. A brief overview of SCM systems that can run on Linux is available.

I've not discussed highly related issues like bug tracking (such as Bugzilla); that's outside the scope of this paper.

Conclusions

The world of OSS/FS SCM systems is a better place than it was a few years ago; there are now several viable options. CVS, while it has its weaknesses, is still a workhorse able to do the basic job. Subversion is ready today for those who just want a better CVS. GNU Arch is extremely capable if you're willing to work with the issues listed above (and it will get better). Personally, although I'd be happy to use subversion on others' projects, I personally plan to use GNU Arch; its warts are numerous, but I think they'll be rapidly fixed and GNU Arch has a tremendous amount of promise. There are other options, too, as discussed above; Monotone in particular looks interesting. I hope you've found this brief tour helpful.

Feel free to also look at my paper on SCM security, or see my home page at http://www.dwheeler.com.

This article is re-printed with permission. The originals can be found at:

http://www.dwheeler.com/essays/scm.html

Book reviews

Author: Martin Schwenke <martins@canb.auug.org.au> , <martin@meltin.net>

Free As In Freedom (Richard Stallman's Crusade for Free Software) by Sam Williams.

"Free As In Freedom" is a biography of Richard Stallman (RMS), the founder of the GNU project, the Free Software Foundation (FSF), author of the GNU Emacs editor, and all-round free software crusader. The book steps back and forth through various stages in the history of Free and Open Source Software (FOSS) and the parts RMS has played in these stages. You get various snippets: "the printer story", RMS as a child, RMS as a young man, RMS as a young hacker, RMS as a disgruntled hacker, impressions of RMS as a public speaker, RMS being weird about women, Emacs, GNU, the GPL, GNU/Linux and Open Source. It's "all" there, and much of it is an interesting read, especially from a historical perspective. However, by the end of the book you feel that, although you know a lot of possibly useful stuff, you don't necessarily have a good overall picture of RMS as a person.

As a long-time Emacs user, and occasional contributor to Emacs and related projects, I've had quite a few interactions with RMS over the years - not all necessarily positive. Apparently I'm not in an exclusive club in this respect - many people are critical of RMS. I've seen RMS speak: he did "the printer story" and "the Saint Ignucius thing" - it was interesting, educational and entertaining. However, I still don't know much about RMS himself. The main thing I wanted to get out the book was an explanation of why RMS is RMS, at both a political and a personal level.

The chapters about RMS as a child and young man are interesting. Much of one chapter is taken up with conjecture about whether or not RMS suffers from a behavioural disorder, in an effort to explain his interesting, sometimes abrasive, personality. However, there is no conclusion either way. In a later chapter we learn that RMS is uncompromising. We learn about communal software projects that take proprietary detours, angering RMS and strengthening his resolve. We learn that he hates anti-social behaviour when it comes to sharing software.

So what don't we learn?

For example, I really wanted RMS's take on the GNU Emacs versus Lucid/XEmacs schism. This might have given some insight into RMS's attitude towards lots of things. After all, this seems to have been something that he has taken personally. However, this whole ongoing incident is relegated to a single paragraph, plus a footnote at the end of a chapter that points to Jamie Zawinki's version of events on his web site.

Near the end of the book, the chapters on GNU/Linux and Open

Source are interesting, but RMS seems to be at the periphery. There's descriptions of his interactions with Ian Murdock and Bruce Perens regarding the Debian project. There's comments about what he think of the term "Open Source", but the chapter focuses on other people. Yet again there are interesting snippets about RMS, but there isn't any depth...

... and I guess that probably brings us to the point...

As well as being an excellent hacker, RMS is political activist. As such, he holds certain beliefs and the book reassures us that he is passionate about them. In doing so we are told a bunch of useful information about RMS's beliefs, but the book depends on narrative rather than analysis to justify them. We don't learn whether RMS loses sleep thinking about "Linux", Microsoft, computer viruses, the term "Open Source"... or global warming. We don't generally get inside RMS's head and, therefore, we don't learn anything terribly interesting about him, apart from the possibility that he's just a single-dimensional Free Software zealot.

I wanted more! I wanted to see the "real" RMS! Perhaps I did - perhaps that's all there is? Oh yeah, he likes to dance...

For those who don't already know a lot about RMS and the Free Software movement, this is probably a good read. Note that the book is also available for free on the web, as it is published under the GNU Free Documentation License. However, I prefer things like this spiffy hardcover, since they're much more convenient to read.

peace & happiness,
martin

Advertisement: American Bookstore

AUUG Chapter Meetings and Contact Details

CITY	LOCATION	OTHER
ADELAIDE	TBA	For updated information, see http://www.auug.org.au/saauug/ or Contact sa-exec@auug.org.au for further details.
BRISBANE	Inn on the Park 507 Coronation Drive Toowong	For further information, contact the QAUUG Executive Committee via email (qauug-exec@auug.org.au). The techno-logically deprived can contact Rick Stevenson on (07) 5578-8933. To subscribe to the QAUUG announcements mailing list, please send an e-mail message to: <majordomo@auug.org.au> containing the message "subscribe qauug <e-mail address>" in the e-mail body.
CANBERRA	Australian National University	For updated information, see http://www.canb.auug.org.au/cauug/
HOBART	University of Tasmania	For updated information, see http://www.tas.auug.org.au/
MELBOURNE	Various. For updated information See: http://www.vic.auug.org.au/	The meetings alternate between Technical presentations in the even numbered months and purely social occasions in the odd numbered months. Some attempt is made to fit other AUUG activities into the schedule with minimum disruption.
PERTH	The Victoria League 276 Onslow Road Shenton Park	For updated information, see http://www.auug.org.au/wauug/waug.html
SYDNEY	Meetings start at 6:15 pm Sun Microsystems Ground Floor 33 Berry Street (cnr Pacific Hwy) North Sydney	The NSW Chapter of AUUG is now holding meetings once a quarter in North Sydney in rooms generously provided by Sun Microsystems. More information here: http://www.auug.org.au/nswauug/

FOR UP-TO-DATE DETAILS ON CHAPTERS AND MEETINGS, INCLUDING THOSE IN ALL OTHER AUSTRALIAN CITIES, PLEASE CHECK THE AUUG WEBSITE AT HTTP://WWW.AUUG.ORG.AU OR CALL THE AUUG OFFICE ON 1-800-625655.

Annual Election of Officers and General Committee Members: Call For Nominations

GET INVOLVED!

AUUG has a proud 29 year history of sharing knowledge, providing member services and, most importantly, creating a community of like minded professionals. Every year brings fresh challenges and new opportunities. As a result, AUUG is in a constant process of evolution; a process of which every member in our association is a part. This year will mark a particularly interesting chapter in AUUG's evolution: for the first time in nearly ten years, we will reevaluate our position in the industry. We expect significant changes as a result.

The role of AUUG's Officers and General Committee Members is to manage, plan and execute, according to the will of the general membership. This stewardship is not passive, nor is it always easy. However, serving the AUUG community is also immensely rewarding because, simply, our goals matter and we can make a difference.

What should AUUG be doing next year? How can we serve our members and our community better? What great ideas are out there, just waiting for their chance to be tried out? How do we better promote our knowledge and philosophies? Do you know the answers to some of these questions? Are you the sort of person who knows how to get things done? Or do you know someone like this? AUUG needs people with fire and clue. Help make AUUG the kind of association you want it to be--nominate the best people for election to our Management Committee.

If you would like to know more about serving on the Management Committee, email the current committee at auugexec@auug.org.au. In order to stand for office, you must be an Individual Member of the AUUG, and you need to be nominated by three voting members of AUUG (that is, either Individual Members or Institutional Members). If you can't find three people to nominate you, send in your nomination form anyway. We should be able to find someone to sign it.

In order to nominate a member for the Committee, please copy and fill out the following official nominationform, and send it to the AUUG Secretary. All nominations must be received by 28 April 2004. You can send in nominations by fax or (snail) mail: Fax: (02) 8824 9522 Mail:

AUUG Inc
PO Box 7071
Baulkham Hills BC NSW 2153 Australia

We encourage nominees to include a policy statement of up to two hundred words. This statement will be circulated to members with election materials, and is intended to assist them in making voting decisions. The Secretary reserves the right to truncate lengthy statements in order to minimise election expenses.

AUUG Inc. 2004 Annual Election: Nomination Form

We,

(1) Name: AUUG Member #: and

_____

(2) Name: AUUG Member #: and

_____

(3) Name: AUUG Member #:

_____

being current financial members of AUUG Inc do hereby nominate:

_____

for the following position(s):
Mark the boxes against the positions for which nomination is desired. Each person may be elected to at most one position, and election shall be determined in the order shown on this nomination form.

1. President
2. Vice President
3. Secretary
4. Treasurer
5. Ordinary Management Committee Member (5 positions)
6. Returning Officer
7. Assistant Returning Officer

Signed (1) _____ Date: _____

Signed (2) _____ Date: _____

Signed (3) _____ Date: _____

I (name):_____ AUUG Member #:_____
do hereby consent to my nomination to the above position(s), and declare that I am currently a financial Individual Member of AUUG Inc.

Signed:_____ Date:_____

Application / Renewal Individual or Student Membership of AUUG Inc.

Use this tax invoice to apply for, or renew, Individual or Student Membership of AUUG Inc. To apply online or for Institutional Membership please use **http://www.auug.org.au/info/**

This form serves as Tax Invoice.

Please complete and return to:

AUUG Inc, PO Box 7071, BAULKHAM HILLS BC NSW 2153, AUSTRALIA

If paying for your membership with a credit card, this form may be faxed to AUUG Inc. on +61 2 8824 9522.

Please do not send purchase orders.
Payment must accompany this form.

Overseas Applicants:

- Please note that all amounts quoted are in Australian Dollars.
- Please send a bank draft drawn on an Australian bank, or credit card authorisation.
- There is a $60.00 surcharge for International Air Mail
- If you have any queries, please call AUUG Inc on +61 2 8824 9511 or freephone 1800 625 655.

Section A:

Personal Details

Surname: .....

First Name: .....

Title: ..... Position: .....

Organisation: .....

Address: .....

.....

Suburb: .....

State: ..... Postcode: .....

Country: ..... Phone Work: .....

Phone Private: ..... Facsimile: .....

E-mail: .....

Membership Number (if renewing): .....

Student Member Certification

For those applying for Student Membership, this section is required to be completed by a member of the academic staff.

I hereby certify that the applicant on this form is a full time student and that the following details are correct:

Name of Student: .....

Institution: .....

Student Number: .....

Signed: .....

Name: .....

Title .....

Date Signed: .....

Section B: Prices

Please tick the box to apply for Membership. Please indicate if International Air Mail is required.

Renew/New* Individual Membership	$110.00 (including $10 GST)	❒
Renew/New* Student Membership	$27.50 (including $2.50 GST)	❒
Surcharge for International Air Mail	$60.00	❒

* Delete as appropriate.

GST only applies to payments made from within Australia. Rates valid from 1st October 2002.

Section C: Mailing Lists

AUUG mailing lists are sometimes made available to vendors. Please indicate whether you wish your name to be included on these lists:

Yes ❒ No ❒

Section D: Payment

Pay by cheque

Cheques to be made payable to **AUUG Inc**. Payment in Australian Dollars only.

OR Pay by credit card

Please debit my credit card for A$ .....

Bankcard ❒ Mastercard ❒ Visa ❒

Card Number: ..... Expires: .....

Name on card: ..... Signature: .....

Date Signed: .....

Section E: Agreement

I agree that this membership will be subject to rules and bylaws of AUUG Inc as in force from time to time, and this membership will run from the time of joining/renewal until the end of the calendar or financial year as appropriate.

Signed: .....

Date Signed: .....

This form serves as Tax Invoice. AUUG ABN 15 645 981 718

Volume 25 ● Number 1
March 2004

FEATURES:

KDE 3.2 Review 7
rsynch The best backup system ever 10
Going 3D with Blender: Modelling a chest 12
Tuxpaint: A paint program for kids 15
2003 & Beyond: Final 19
StoreBackup 28
Programmers toolkit: Profiling programs using gprof 32
Certs for the Masses 34
Cyberinsecurity: The Cost of Monopoly (Part 2) 37
Bock Review: Practical VoIP 41
Overheard in the office 42
History of the transport of computer viruses via email 43
Book Review: Just for Fun 45
Book Review: The Complete FreeBSD 45

FreeBSD 5.2 Review 46
Book Review: Perl for Oracle DBAs 49
Book Review: Tomcat – The Definitive Guide 50
Comments of OSS/FS Software Configuration Management Systems 50
Book Review: Free as in Freedom 55

NEWS:

Public Notices 4
AUUG Conference 2004: Call for Papers 18
AUUG: Corporate Members 31
AUUG: Annual Election of Officers 59
AUUG: Chapter Meetings and Contact Details 58

REGULARS:

President's Column 3
My Home Network 4